Design • Build • Program • Learn • Compete

SERVO MAGAZINE

FOR THE ROBOT INNOVATOR

www.servomagazine.com

November 2013

Human AUGMENTATION

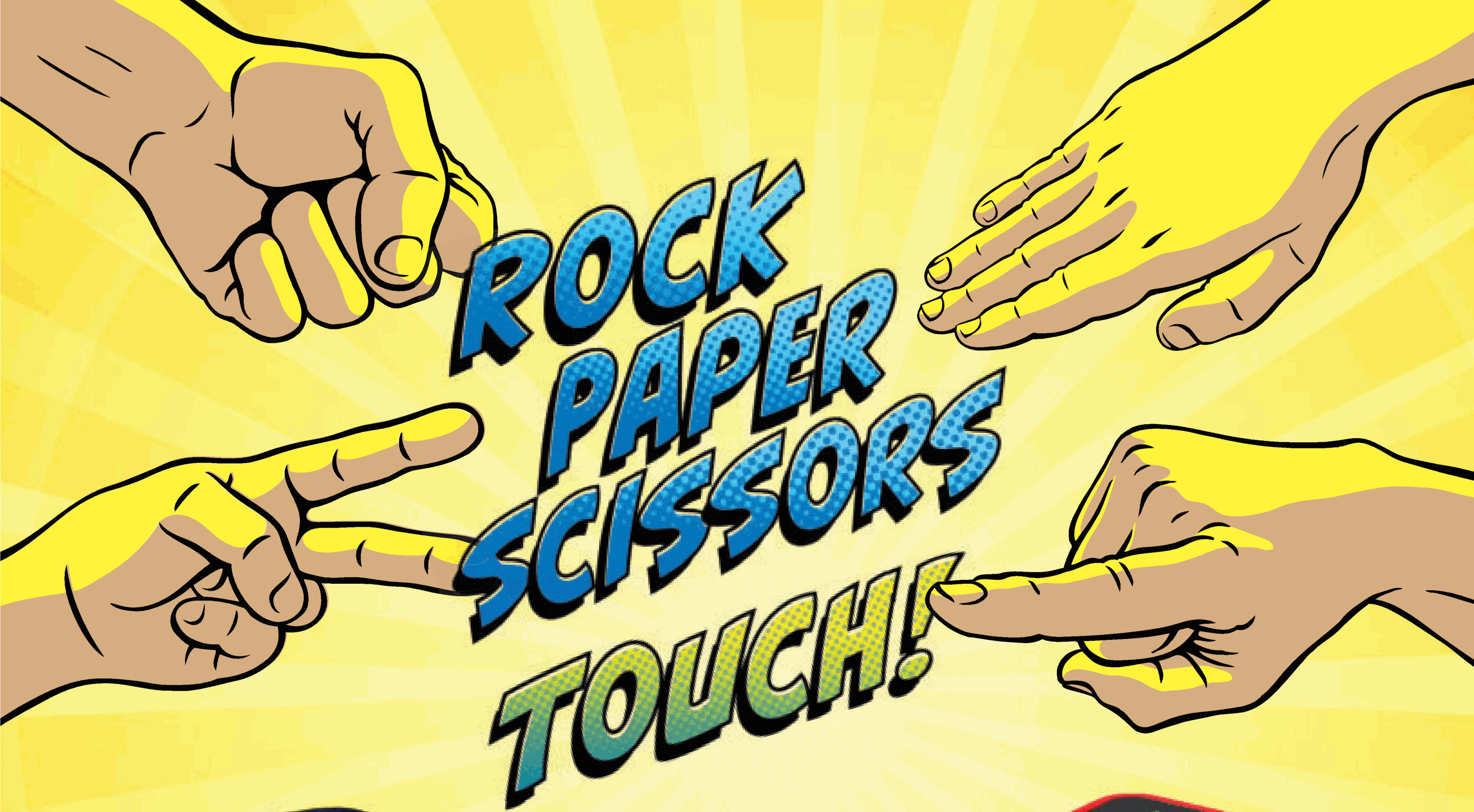
Hollywood vs Reality

How Far Have We Come?

U.S. $5.50 CANADA $7.00

11

0 71486 02422 4



X1 TOUCH 200 DC | DC

DC/DC Input, Professional Balance Charger, Discharger

X1 TOUCH AC | DC

AC/DC Input, Professional Balance Charger, Discharger

Charging your batteries is just a simple touch away with the X1 Touch and X1-200 Touch multi-chemistry chargers. Both feature a 3.2-inch, high-resolution touch screen and will efficiently charge your NiMH, NiCd, LiPo, LiFe, Li-Ion and lead-acid batteries. The X1 Touch is equipped with a built-in AC/DC power supply, 55-watt charge port and internal balancing circuit. If you need more power, the DC/DC X1-200 Touch features 200 watts of charging power and optimized operating software. Get the Hitec touch and let your fingers do the charging!

Nothing Beats a Perfect Charge!

HITEC

Hitec RCD USA Inc. / 12115 Paine St. Poway, CA 92064 / (858) 748-6948 / www.hitecrcd.com


Pololu
Robotics & Electronics
Finding the right parts for your robot can be difficult, but you also don't want to spend all your time reinventing the wheel (or motor controller). That's where we come in: Pololu has the unique products — from actuators to wireless modules — that can help you take your robot from idea to reality.
170-Point Breadboard
ITEM #1486
$2.95
Other colors available
• 17 rows of tie points for up to two 14- or 16-pin DIP ICs
• Tabs allow multiple units to be connected for larger projects
Wall Power Adapters
STARTING AT
$5.95
• Center positive, 5.5 mm × 2.1 mm barrel connector
• Available voltages: 5 V, 9 V, and 12 V
• Available currents: 1 A, 3 A, and 5 A
Dual Motor Drivers
ITEM #2135
$4.49
• DRV8835 driver
• 2 V to 11 V operating range
• 1.2 A continuous (1.5 A peak) per channel
• 14-pin DIP form factor
ITEM #713
$4.95
• TB6612FNG driver
• 4.5 V to 13.5 V operating range
• 1 A continuous (3 A peak) per channel
ITEM #2130
$4.95
• DRV8833 driver
• 2.7 V to 10.8 V operating range
• 1.2 A continuous (2 A peak) per channel
Pololu Basic 2-Channel SPDT Relay Carrier with 12 VDC Relays
ITEM #2487
$8.25
Control two SPDT relays easily with logic level signals. Available with 5 V relays or without relays. Single carrier also available.
Linear Actuators
STARTING AT
$66.95
Various sizes available
• 2" to 12" stroke lengths
• Multiple speeds available
• Potentiometer feedback option
QTR-L-1RC Reflectance Sensor
ITEM #2455
$4.95/2-pack
Add line sensing or proximity/edge detection to your robot with these tiny sensors, capable of measuring infrared reflectance at distances of up to 1 inch.
Addressable RGB 30-LED Strip, 1 m
ITEM #2543
$34.95
Waterproof, individually addressable RGB LED strip that runs on 5 V. Can be chained to form longer strips or cut for shorter sections. Also available in 2 m and 5 m lengths.
Jrk 21v3 USB Motor Controller
ITEM #1392
$49.95
Highly configurable motor controller that offers four control interfaces and can optionally be used with feedback for closed-loop speed or position control. Great for use with our linear actuators or making your own servo!
m3pi Robot
ITEM #2153
$189.95
3pi upgrade kit available
Enhanced version of our popular 3pi robot. Uses the 32-bit mbed development board to offer more processing power and free I/O lines.
Optical Encoder Pair Kit for Micro Metal Gearmotors
ITEM #2590
$8.95
Compatible Gearmotors
$16.95
Extends only 5 mm past motor
Add quadrature encoding to your micro metal gearmotors!
• 3- and 5-tooth wheels included
• 3.3 V and 5 V versions available
• Works with our growing assortment of gearmotors with extended back shaft (gearmotors sold separately)
Dagu Wild Thumper 6WD Chassis (With 75:1 Steel Gearboxes)
ITEM #1563
• Rugged aluminum body
• All-terrain suspension
• Available in 4WD and other gear ratios and colors
$249.95
Dagu Rover 5 Tracked Chassis
ITEM #1550
$49.95
• Durable rubber tracks
• Adjustable ground clearance
• Available with encoders
5" Robot Chassis RRC04A
• Laser-cut acrylic base plate
• Many colors available
• Add motors, wheels, a ball caster, and a controller to build a complete robot
$7.95
Find these products and more at: www.pololu.com

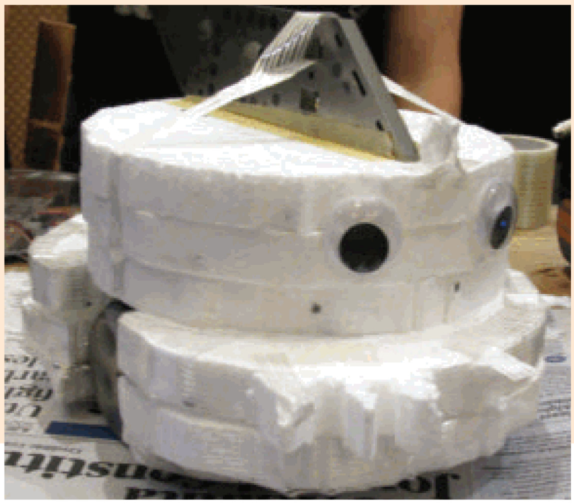
SERVO MAGAZINE

11.2013

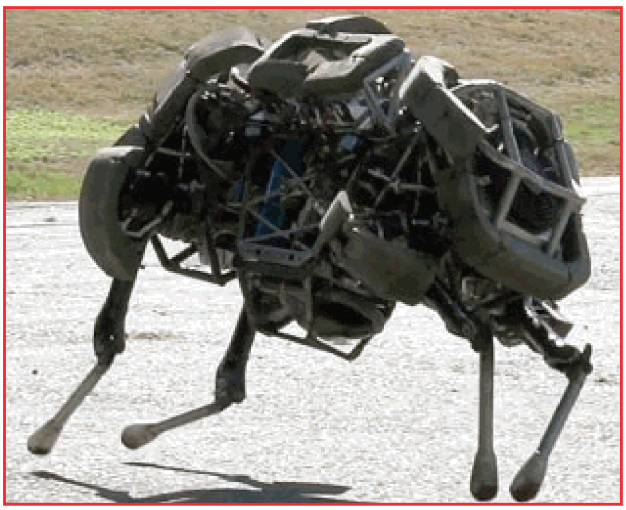
VOL. 11 NO. 11

Columns

The Combat Zone...

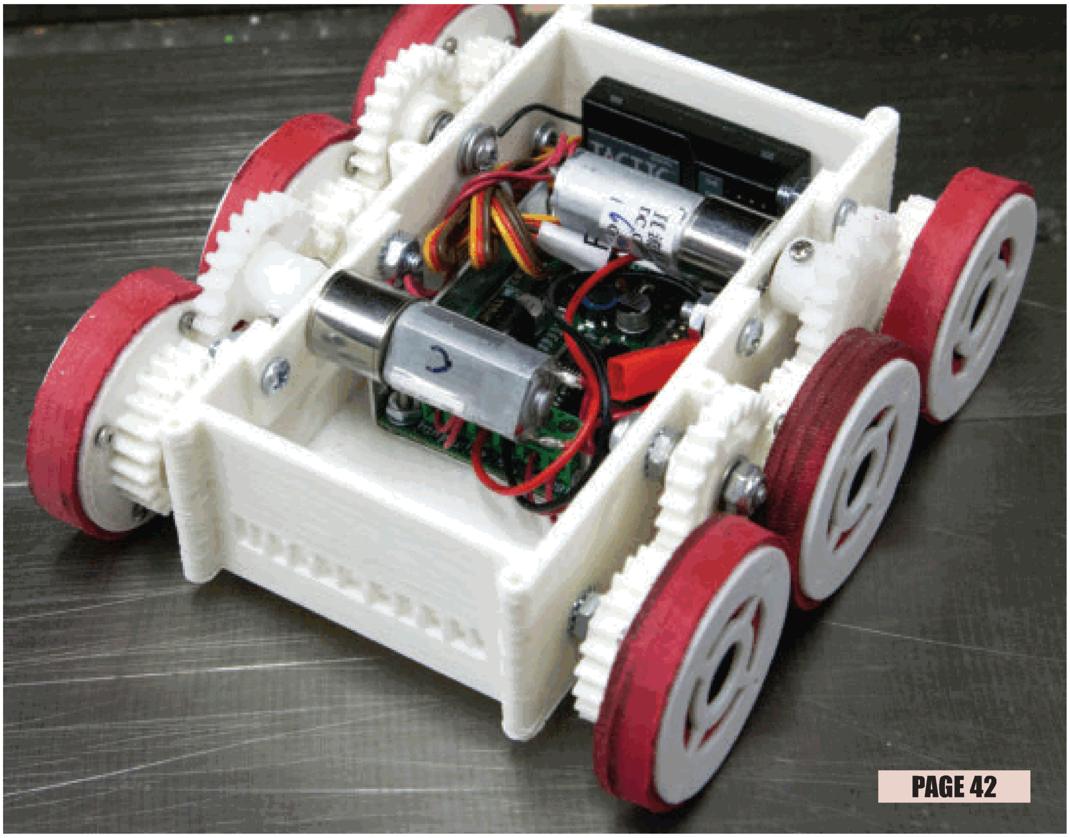


Departments



SERVO Magazine **(ISSN 1546-0592/CDN Pub Agree#40702530)** is published monthly for $24.95 per year by T & L Publications, Inc., 430 Princeland Court, Corona, CA 92879. PERIODICALS POSTAGE PAID AT CORONA, CA AND AT ADDITIONAL ENTRY MAILING OFFICES. POSTMASTER: Send address changes to **SERVO Magazine, P.O. Box 15277, North Hollywood, CA 91615** or Station A, P.O. Box 54, Windsor ON N9A 6J5; cpcreturns@servomagazine.com

In This Issue ...



Mind / Iron

by Bryan Bergeron, Editor

Biomimetics and Perception

Biomimetics — also known as biological mimicry — has contributed to robotic design since at least the time of William Walter's turtles in the 1950s. Leonardo da Vinci's design for flying suits that mimicked the wings of birds is probably the first technological application of biomimetics. Unfortunately, material science wasn't advanced enough to support his visions.

The same could be said for modern robotics. We haven't quite figured out how to construct a fully deformable platform akin to the T-1000 liquid metal robot in Terminator 2. Certainly, there are material science and computer science issues involved. However, I think that the greatest challenges to creating a robot that could pass for a human are the perceptual issues that are less obvious than, say, how to mimic the pseudopod generation in the amoeba.

Why build something that could pass for a human or a robot with one or two human traits? Take enabling a search and rescue robot to identify and localize the sounds of victims trapped under the rubble of a collapsed building. Seems straightforward enough, right? Simply mimic the physiology and to some extent, the anatomy of the human ear. There's clearly practical value in such a robot, if it could be constructed.

Let's start with the basic sound localization specifications of the human ear. It's well known that the human ear is sensitive to the relative amplitude and phase of acoustic vibrations. Furthermore, the directional characteristics of our external ears modify the vibrations reaching each ear — especially audio frequencies less than about 6 kHz.

Another factor that contributes to our ability to localize sounds is the equivalent of sensor fusion from multiple sense organs. Auditory cues are combined with information from the position and movement sense organ in the ears, eyes, and motion, and position sense organs in the muscles, tendons, and joints. To get a sense for this sensor fusion in action, consider the automatic reflex action of rotating the head from side to side to better localize the source of a sound. The resulting variation in the relative amplitude and phase relationships of signals reaching the ears provides the auditory system with additional data points that are used to more accurately localize the signal source.

It's easy enough to mimic these capabilities. I've done so with a microcontroller, a few directional microphones, and a few additional sensors. While the system is useful in localizing sounds, the results don't match those of a human. Why? It turns out that several properties of the human auditory system defy explanation on a strictly physiological or anatomical basis, but are instead best understood in terms of human perception of sound or psychoacoustics.

The psychoacoustic property most applicable to localization is perceived intensity. The perceived intensity of a sound is a function of the audio signal's duration. While sounds that last longer than about 250 ms and are of equal amplitude, they are perceived as having equal intensity; shorter duration sounds of the same amplitude are perceived to have a lower intensity. Quantitatively, a decade increase in duration, say, from 50 ms to 500 ms, is equivalent to a 10 dB increase in intensity — as long as it involves crossing the 250 ms threshold. There are other psychoacoustic properties that don't directly affect our ability to localize sounds. For example, through conditioning, some sounds are pleasant and others are annoying.

So, what's the practical take-away from this minutia about human hearing? The point is that you can't limit your mimicry to the system you're studying. If your goal is to duplicate human capabilities — whether in vision, hearing, touch, or smell — don't forget to include the perceptual components of the system you're attempting to mimic. It's easy enough to model the effects of sound duration on perceived intensity — once you know that they exist. SV

FOR THE ROBOT INNOVATOR SERVO MAGAZINE

Published Monthly By
T & L Publications, Inc.
430 Princeland Ct., Corona, CA 92879-1300
(951) 371-8497
FAX **(951) 371-3052**
Webstore Only **1-800-783-4624**
www.servomagazine.com

Subscriptions
Toll Free **1-877-525-2539**
Outside US **1-818-487-4545**
P.O. Box 15277, N. Hollywood, CA 91615

PUBLISHER
Larry Lemieux
publisher@servomagazine.com

ASSOCIATE PUBLISHER
Robin Lemieux
robin@servomagazine.com

EDITOR
Bryan Bergeron
techedit-servo@yahoo.com

CONTRIBUTING EDITORS
Jeff Eckert, Jenn Eckert, Tom Carroll, Kevin Berry, Dennis Clark, R. Steven Rainwater, Michael Simpson, Gordon McComb, Fred Eady, Mike Jeffries, Dan Albert, Chris Mayer, Girts Linde, Alan Schilling, David Geer, Dave Calkins

ADVERTISING SALES EXECUTIVE
Richard Lockhart
richard@servomagazine.com

CIRCULATION DEPARTMENT
subscribe@servomagazine.com

MARKETING COORDINATOR WEBSTORE
Brian Kirkpatrick
sales@servomagazine.com

WEB CONTENT
Michael Kaudze
website@servomagazine.com

ADMINISTRATIVE ASSISTANT
Debbie Stauffacher

PRODUCTION
Sean Lemieux

Copyright 2013 by
T & L Publications, Inc.
All Rights Reserved

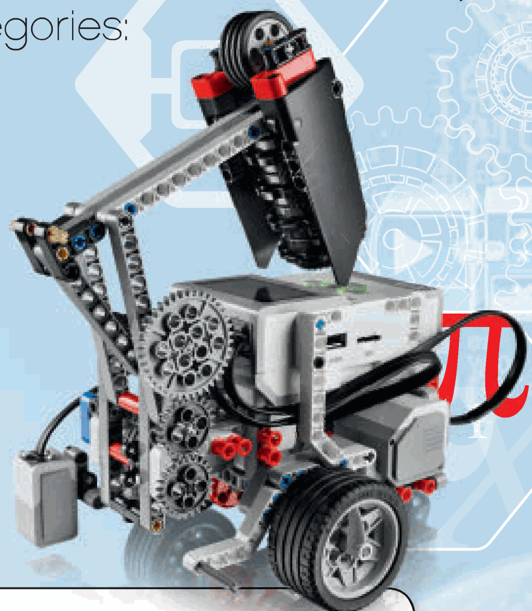
All advertising is subject to publisher's approval. We are not responsible for mistakes, misprints, or typographical errors. *SERVO Magazine* assumes no responsibility for the availability or condition of advertised items or for the honesty of the advertiser. The publisher makes no claims for the legality of any item advertised in *SERVO*. This is the sole responsibility of the advertiser. Advertisers and their agencies agree to indemnify and protect the publisher from any and all claims, action, or expense arising from advertising placed in *SERVO*. Please send all editorial correspondence, UPS, overnight mail, and artwork to: **430 Princeland Court, Corona, CA 92879.**
Printed in the USA on SFI & FSC stock.

TEACH TOMORROW'S ROCKET SCIENTISTS WITH LEGO® MINDSTORMS® EDUCATION EV3 SPACE CURRICULUM

The EV3 Space Challenge Set & Activity Pack is comprised of three learning mats, one challenge mat, a large number of LEGO elements, and 30+ hours of activities in the following categories:

Basics of Gears

Students apply physical science and math principles to build effective robots.



Learning Missions

Students progress through nine distinct missions in which they investigate, observe, calculate, and apply their knowledge to solve specific tasks.

Research Projects

Students explore, plan, and develop around three fundamental challenges NASA engineers and scientists are working to solve – how to ensure humans can survive in space, how humans can create energy in space, and how robots can help humans explore space.

Challenge Missions

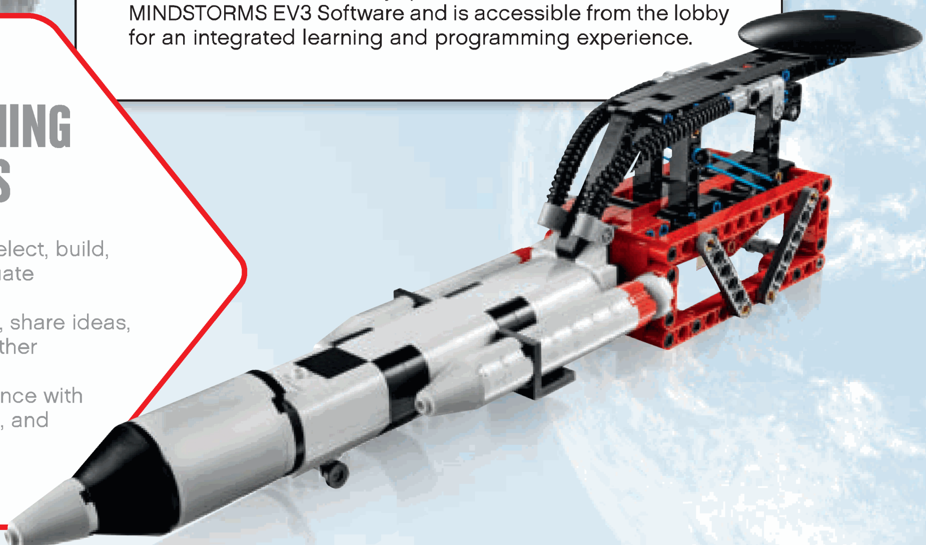
Students apply and creatively adapt programming and problem-solving skills to design and build robots to solve seven very different space challenges.

Introduce your students to robotics with a very current and real-world theme – traveling to and living on Mars! Students will work as scientists and engineers as they progress though this set of lessons, challenges, and projects that fully integrate science, technology, engineering, and math concepts while prompting creative problem solving, communication, and teamwork.

This multimedia activity pack installs into the LEGO MINDSTORMS EV3 Software and is accessible from the lobby for an integrated learning and programming experience.

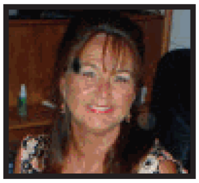
KEY LEARNING VALUES

- Develop solutions, select, build, test, and evaluate
- Learn to communicate, share ideas, and work together
- Hands-on experience with sensors, motors, and software



LEGOeducation.us/MINDSTORMS

LEGO mindstorms education EV3

Robytes

by Jeff and Jenn Eckert

VTOL Anything

Since 1943, the famous Skunk Works (a.k.a., Advanced Development Programs) division of Lockheed Martin (**www.lockheedmartin.com**) has been responsible for many advanced (and secretive) aircraft designs, including the U-2 spy plane, the F-117 Nighthawk, and the F-35 Lightning II. More recently, the skunks have been working with Piasecki Aircraft (**www.piasecki.com**) to develop a new autonomous vehicle dubbed the Transformer TX.

The original concept (as described back in the January 2011 issue) was to build a rotor-driven flying car, but eventually the designers decided to dump the car portion of it, thereby creating a mechanism that can lock onto any suitable cargo pod and fly away with it. The production version — scheduled to take flight by 2015 — will be powered by ducted fans that can drive it at 200 knots within a range of 250 miles.

According to the company, the TX's unique design will allow it to "adapt to multiple missions with interchangeable payloads, including cargo pods, medical evacuation units, a tactical ground vehicle, armed scouts, and reconnaissance and strike capabilities."

Lockheed Martin's Transformer TX can turn virtually anything into a VTOL vehicle.

Lots of Pickin', No Grinnin'

It is generally accepted that we have moved past the age of the lone inventor working in his garage, but an exception to the rule seems to be Vladimir Demin — a 62-year-old circuit designer currently residing in Moscow. Using nothing but hand tools (primarily a drill, as the project required 3,000+ high-tolerance holes), he created an automatic guitar-playing machine.

The device uses a slew of solenoids for picking and fretting — all controlled via an ADSP2187 CPU. It draws 30A at 12V, but Vladimir says it can play continuously for two to four hours without a battery change. Fortunately, Eric Clapton doesn't need to worry about the competition, but it does do a pretty amazing job on an old Russian song that must remain unnamed here as I don't seem to have a Cyrillic font on my machine. You can check out the video by searching on YouTube for his name. His next project — in case you are wondering — will be a high-tech accordion player that can perform a duet with the guitar.

Vladimir Demin's automatic guitar.

Go to **www.servomagazine.com/index.php?/magazine/article/november2013_Robytes** to comment on these topics.

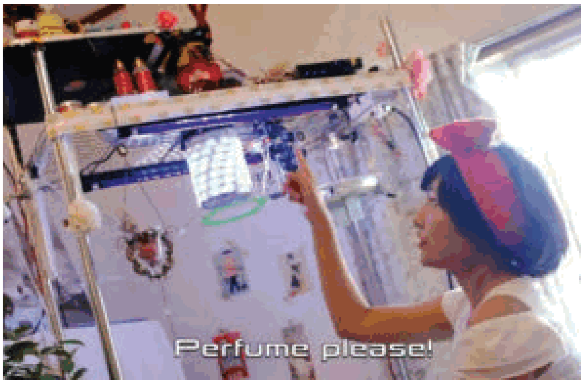
ETF Indexes Automation Industry

New Robo ETF designed to track automation stocks.

Even if you aren't a major player in the investment market, the term ETF (exchange traded fund) may be familiar to you. If not, it is sufficient to note that an ETF is a type of mutual fund that is traded like stock. It is like an index fund in that it is linked to some slice of a particular industry, and is intended to track gains in that (hopefully booming) industry. Which brings us to an interesting new ETF that was recently filed with the Securities and Exchange Commission. Called the Robo-Stox Global Robotics and Automation Index ETF, it is the brainchild of J. Garrett Stevens, CEO of Exchange Traded Concepts, LLC (**www.exchangetradedconcepts.com**).

The Robo ETF is scheduled to be listed on the Nasdaq exchange by the time you read this. Details are sketchy, but it has been revealed that the fund will be putting at least 80 percent of its investments in companies that "derive a significant portion of revenues and profits from robotics or automation-related products or services." The fees are relatively high at 0.95 percent annually (compared to the 0.53 percent industry average), but if you want to hitch your financial wagon to the automation industry, you might ask your favorite stockbroker about this.

Razing a Stink



Frame from a video showing the FragWrap robot in action.

This month's suggestion for the Rube Goldberg Machine Contest goes to engineers at Japan's Keio University (**www.keio.ac.jp**) who have come up with a voice-activated air freshener known as "FragWrap" (short for "Fragrance-Encapsulated and Projected Soap Bubble for Scent Mapping").

The robot asks you to choose a fragrance, after which it sucks up the appropriate liquid from a rack of smell sources and squirts it into the bubble generator. The generator produces a fog-filled soap bubble using a large Wonder Bubbles-style wand, a fan, and electric induction. Finally, the bubble is projected downward and popped, filling the room with your favorite fragrance. Okay, with the colored lights and all that, it's entertaining, and maybe Barbra Streisand would buy one. Most of us will just stick with scented candles or a can of Air-Wick. See a video at **vimeo.com/69445335**. **SV**

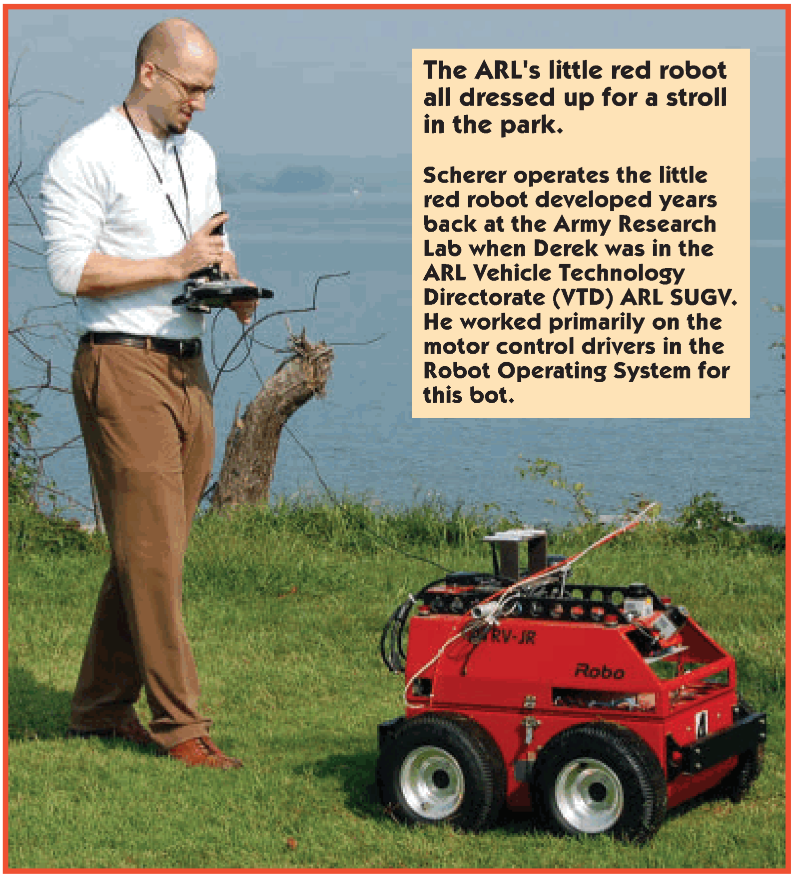
by David Geer

Contact the author at geercom@windstream.net

The Famous Robots of Derek Scherer

From working on Army robots to creating animatronics that star in blockbuster films, roboticist Derek Scherer has come a long way in six years. Check out Scherer's robots from the past and see whether you can spot his recent creations in movies such as "The Hobbit," "Man of Steel," and "Elysium."

That's right. You can graduate from top-secret Army robotics projects to making one-of-a-kind animatronics for some of the biggest movies to come out of Hollywood. Here's Scherer's story.



The ARL's little red robot all dressed up for a stroll in the park.

Scherer operates the little red robot developed years back at the Army Research Lab when Derek was in the ARL Vehicle Technology Directorate (VTD) ARL SUGV. He worked primarily on the motor control drivers in the Robot Operating System for this bot.

Little Red Robot/Big (and Small) Bad Movie Bots

Through 2007, the modest Scherer worked on platforms such as what he calls the Army Research Laboratory's (ARL) little red robot. The product of much earlier projects, the little red robot survived to be a platform for increasing robotics capabilities with testing and development.

Scherer worked on the motor drivers — software drivers for a given motor controller board, to be exact. To do that, he had to learn Python the hard way: in real time as he advanced the driver development tasks. The driver Scherer was improving already existed in the Python language, so it was easier to learn Python on the job to move the project forward than to start over in another language that he already knew.

The little red robot used a Roboteq OTS motor controller board for large AC motors with rotary encoders. Though Scherer wrote his Python-based driver code in a hurry, he found that developing script-level code for character robots was a good feature to add to these types of mechanisms.

"In the Robot Operating System (ROS), this script-level accessibility is provided through the high-level Python programming language," he explained

The little red robot was more than 10 years old when Scherer began to work on it as a development platform. "It wasn't very functional."

Typical applications for the robot included maneuvering populated rooms based on heuristically

Post comments on this article at **www.servomagazine.com/index.php?/magazine/article/november2013_GeerHead.**

The Army's XUV robot leading the way in a staged recon mission.

Here, the XUV reconnaissance robot on which Derek Scherer performed testing and assessments appears to be running for its life from friendly fire. While much of what the XUV did is confidential, it served in part as a swift, sharp mobile photographer, taking pictures of potential enemy encampments.

processed sensor data, a priori data, relations, and errata.

A replacement platform would have been too expensive. "While we often think of the government as spending money like mad — and having had an inside view I can say that is sometimes the case — there wasn't enough enthusiasm to get a budget for a multi-hundred thousand dollar robot," Scherer commented.

The XUV Robot From the ARL

While at the ARL, Scherer worked on other robots including the XUV (experimental unmanned vehicle). The ARL used the XUV for developing technologies for reconnaissance missions. The XUV specifically tests the feasibility of autonomous mobility for unmanned vehicles to work in tandem with manned vehicles in deployment. "The robot would drive around the woods and take pictures of possible enemy locations. This research is vital to bringing robots into regular scenarios with people," says Scherer.

The XUV applies a specially developed sensor package and an Army operator remotely controls the robot. The remote control device uses an interface with a map display for the area, mission analysis, planning, and execution tools, and displays and controls for acquiring reconnaissance and surveillance targets.

The XUV also affords soldiers the opportunity to experience working in tandem with autonomous robotic vehicles before being deployed with them. Roboticists like Scherer receive feedback from these XUV field experiments about the human element and soldier's needs in these kinds of deployments. The XUV helps the Army to develop autonomous mobile technology for successful use on a variety of terrains.

"I worked in command and control which considered how battlefield commanders would maintain situational awareness while giving orders to multiple units, potentially including robotic assets. One of the most important aspects of integrating robots into the battlefield is to ensure they do not get in the way of the soldiers. Robots being introduced to a social system need to unequivocally make our lives easier or they'll be rejected," Scherer pointed out.

Resources

Info about Derek Scherer
http://derekscherer.com/about.html

Derek Scherer's Golem Workshop
www.golemworkshop.com/projects.html

Public information on the Army's XUV
www.gdrs.com/robotics/programs/program.asp?UniqueID=19

Movies Derek Scherer worked on:
The Hobbit — **www.thehobbit.com/index.html**
Man of Steel — **http://manofsteel.warnerbros.com/index.html?home**
Elysium — **www.imdb.com/title/tt1535108**

Making Robots for Movies

As mentioned, Scherer has worked on robots that appeared in films such as The Hobbit, Man of Steel, and

Scherer and two of his colleagues at the Army Research Lab examine the technology inside the little red robot. Technologies internal to the robot include a Roboteq OTS motor controller board, large AC motors with rotary encoders, Wi-Fi, and programming using Python.

Elysium. Unfortunately, he is not free to share specific details or any still photography at this time. Of course, there is nothing to stop you from seeing the movies and watching for the robots yourself.

Scherer created mechanical looking things that use electronics to light up and/or move for Elysium and Man of Steel. "Both of these mechanical props were interesting as a change of pace from the organic creatures of The Hobbit. It's a great joy in engineering that we can learn something in one domain and find that it has natural applications in another," Scherer observed.

"I worked on monsters in The Hobbit. Computer Graphics (CG) replace most of these creatures, but there are a few rare treats in the film where you get to see some rubber skin — the practical animatronic effects that I worked on," Scherer noted.

According to Scherer, in the world of character robots, the character comes first. Robotics is simply the enabling technology. It is the same for effects where the monster or prop has to sell on camera.

"For that reason, development begins in the art departments. Those teams develop sizes, colors, personalities, styles, and more. Molds are made of sculptures and castings are handed off to animatronics where we figure out how to bring the pieces to life with electromechanical parts," explained Scherer.

Scherer instilled the movie robot's capabilities through the mechanical core of the robots. "This is a feature of animatronics that I think is ideal for incorporating into designs for working character robots."

"A thin, rigid plastic membrane keeps the skin in place and in the correct shape. We use paddles to push the skin around as it rests on the mechanical core.

A paddle is any shape that is the interface between the actuator (e.g., servomotor) and the skin. Consequently, the paddle often nestles into the mechanical core and moves in relation to that mechanical core," Scherer described.

Can You Make a Robot Film Star?

If you have seen many of the sci-fi, action, and graphic novel-based films of late, you know that many popular films use robotics. Do you have an interest in creating robots for film? SV


THEY SAY THE FUTURE DOESN'T FIT IN A BOX.
Tuesday, January 7
through Friday, January 10, 2014
Las Vegas, Nevada • CESweb.org • #CES2014
GOOD THING LAS VEGAS IS FAR FROM SQUARE.
Over four days, those who shape the future gather in a city built on reinvention. Here, brands, markets and economies converge in what's far more than a tradeshow. And in 2014, there's more opportunity than ever to connect with those who matter. The only question now, why wouldn't you fit it in your future?
Register today at CESweb.org.
2014 International CES
THE GLOBAL STAGE FOR INNOVATION
PRODUCED BY CEA

ASK MR. ROBOTO

by Dennis Clark
Our resident expert on all things robotic is merely an email away.
roboto@servomagazine.com

Tap into the sum of *all human knowledge* and get your questions answered here! From software algorithms to material selection, Mr. Roboto strives to meet you where you are — and what more would you expect from a complex service droid?

Post comments on this article at **www.servomagazine.com/index.php?/magazine/article/november2013_MrRoboto.**

As I am writing this, I find that I'm in my usual position when it comes to getting ready for a competition. That is, I've delayed until the last minute and now find that I'm scrambling to get things ready. Ah, the life of a busy tech geek! There is always something else to work on! Here I am, wrenching away on a robot that I need to finish when ... "squirrel!" (Those who have seen the movie, *UP* will understand this reference.) I didn't get any questions this month, so I'll tell you about one of my other diversions.

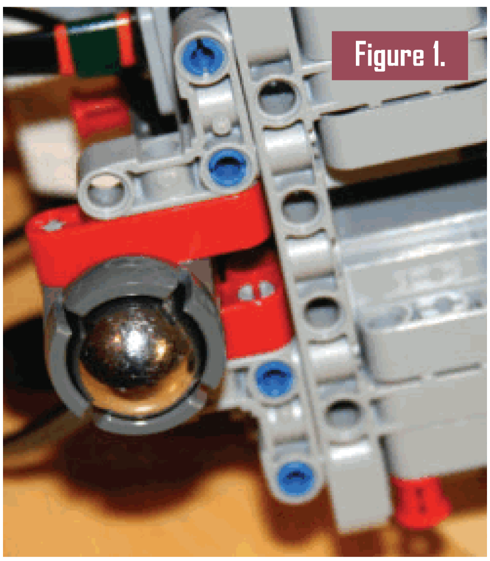
One of the many things keeping me busy this season is the new FIRST LEGO League competition for 2013 — Nature's Fury! Coincidentally, FLL came up with a highly topical theme this year. Here in Northern Colorado, we are dealing with the aftermath of the worst flooding we've seen in years. It has been called our "hundred year flood" event. No matter what it's called, the sheer power of unleashed water and its destructive force was quite a bit to take in for those of us that were in the middle of it! I am reminded once again that man is pretty small change alongside Mother Nature when she decides to let loose. Anyway, back to the FLL and robots.

My team was fortunate to get a first release of the new LEGO MINDSTORMS EV3 robot to compete with. Wow! LEGO listened to much of what we fanatics have been asking for over the last couple of years. The big servomotors have been squared off and have more mounting hole options that allow them to line up better with other building elements, allowing a more compact building footprint (more on that later). Along with the large higher torque motors, the kits now come with a smaller, faster (but less torque) motor whose axle direction is in line with the length-wise axis of the motor.

This allows a smaller motor that fits nicely into spaces that the bigger motors don't. Speaking of motors, we finally get four servomotor connections instead of the time-honored three that the RCX and NXT had. Woo hoo! I have saved the best (in my opinion) for last, however: Finally, we have a true LEGO ball-castor for our skid-steering robots! Take a look at **Figure 1** to see what it looks like on a model.

Add to this some new framing elements, and you can build remarkably useful robots in a small footprint (**Figure 2**). The robot in **Figure 2** is one that you can build by following the kit's manual if you stuff every robot shown together at the same time.

It uses the two large servomotors, the medium servomotor, sonar, and color, touch, and gyro sensors. It is a



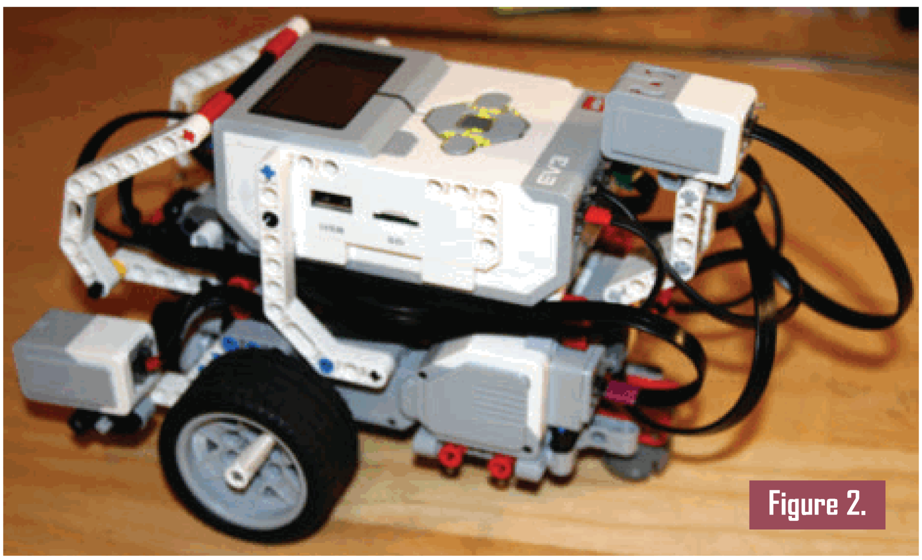


Figure 2.

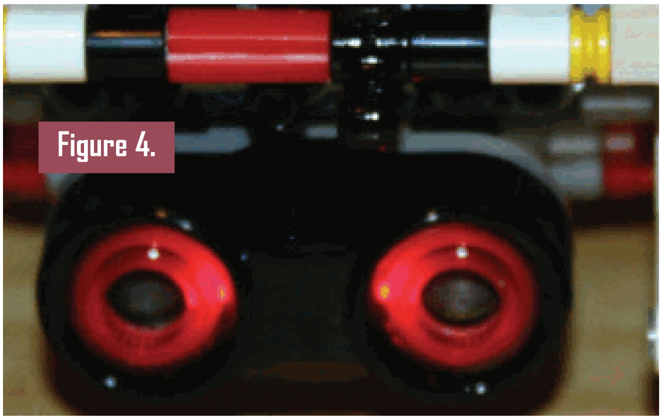


Figure 4.

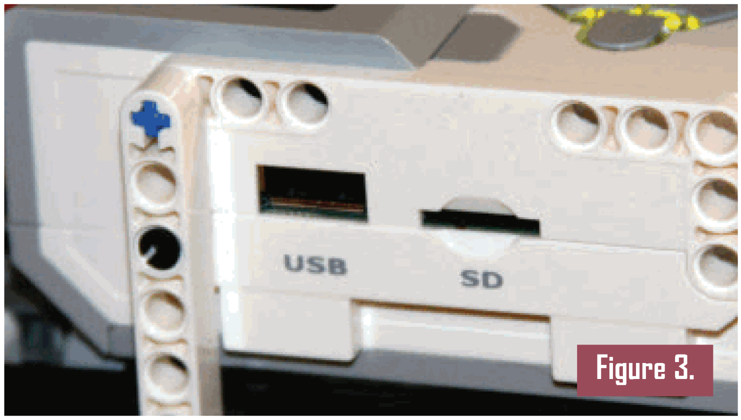


Figure 3.

remarkably compact LEGO robot.

You can see in **Figure 2** that the brain of the EV3 is basically the same overall size, with the same mounting hole locations. It is more boxy and square with fewer rounded edges. I think this is because there is more cool stuff inside it. The programming USB connector isn't the monster size any more; it is a USB mini-B connector placed alongside the more tightly spaced four motor ports.

Check out **Figure 3**. The EV3 has its own USB port and micro-SD card slot. The only thing that LEGO admits to using this USB port for is daisy-chaining up to four EV3s for programming and development, and for attaching a Wi-Fi dongle. (Of course, hackers and fans will find other uses for that port.)

The micro-SD card can hold up to 32 GB, and programs can both read and write to this card for data logging, and perhaps even your own sounds and display graphics (I am not sure about the latter; the documentation is not clear to me on this.)

There is much to learn about the new EV3. I've barely scratched the surface! The new brick OS is embedded Linux on an ARM9 processor, so I look forward to new hacker tools based on gcc to write our own apps without the graphical interface which (I might add) is way nicer looking and seems to be snappier performing too.

When an FLL team gets a MINDSTORMS kit, it is the educational version which is different from the retail version. With the EV3, you get some different building elements and a sonar sensor (see **Figure 4**) which lights up when it's active — which is kind of cool. However, you don't get the new IR sensor or IR remote.

The other really nifty sensor you get with the educational model is the gyro sensor. This is a single axis gyro that apparently can be sampled fast enough to build a classic inverted pendulum robot LEGO calls "Gyro Boy." I've used this sensor to improve dead-reckoning programs by more precisely measuring actual turn angles (**Figure 5**).

The educational model is a little cheaper than the retail unit, but you have to buy the EV3 software separately which makes it more expensive than the retail model. However, the EV3 educational software comes with coursework and helper modules that allow students to embed written documents as well as videos into their projects. It is a very, VERY slick program that I've barely peeked under the hood at!

The new programming blocks are nicer looking and the program "rails" are gone. You can have multiple programs that are not connected, running at the same time. The block connections are cleaner and what I've seen of the data wires leads me to hope that they work better too. What few programs I've run seem to download quicker, and programs seem to respond to sensor inputs much faster than with the NXT.

I have not seen the retail model software, so I can't comment on what that might look or operate like, but I like what I've seen with the educational version. Programs are now written as part of a project and stored on the EV3 in their project's folder. This allows you to do a better job of organizing your projects and programs logically. This is especially important to an FLL team.

Figure 6 shows a simple program that uses a Move Steering block to drive in a circle until the sonar sensor detects something less than 25 cm away. Then, the Loop Interrupter block cancels the "Main" loop and forces an exit, after which the Sound block plays "Goodbye."

You could have put the loop exit test at the end of the Main loop to sense when to exit, but I wanted to show the Loop Interrupter block. This

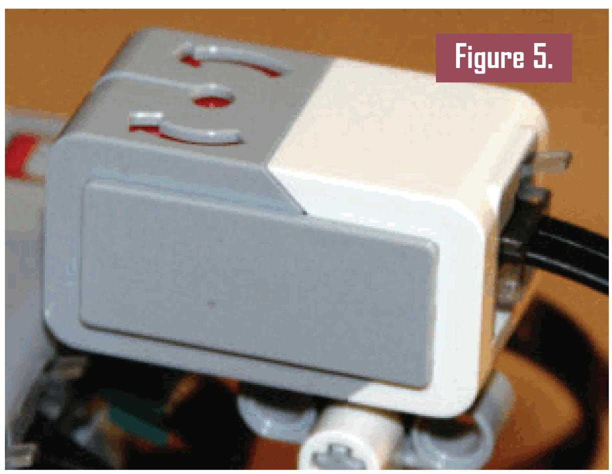


Figure 5.

allows sensors being used in a totally different thread to interrupt any loop in any thread running — not just the one that is running in that thread. This is a very powerful addition to the MINDSTORMS programming toolbox.

I haven't found out if I can start a thread from a decision inside another thread, yet. By the way, you can still use your NXT sensors on the EV3, and the new EV3 environment can also be used to program your "old" NXT, as well. The EV3 robotics environment is a very welcome addition to the truly inspired LEGO MINDSTORMS family.

You'll see that my programming environment is called the EV3 Teacher edition. For obvious reasons, I am trying to teach programming to my group of 9-12 year olds who — by the way — really love the new EV3 programming environment. I've put the retail EV3 on my Christmas "wish list" to see if I can get my "significant other" to get me one. Then, I'll be able to compare the two programming environments to see what LEGO gives to the non-educational crowd.

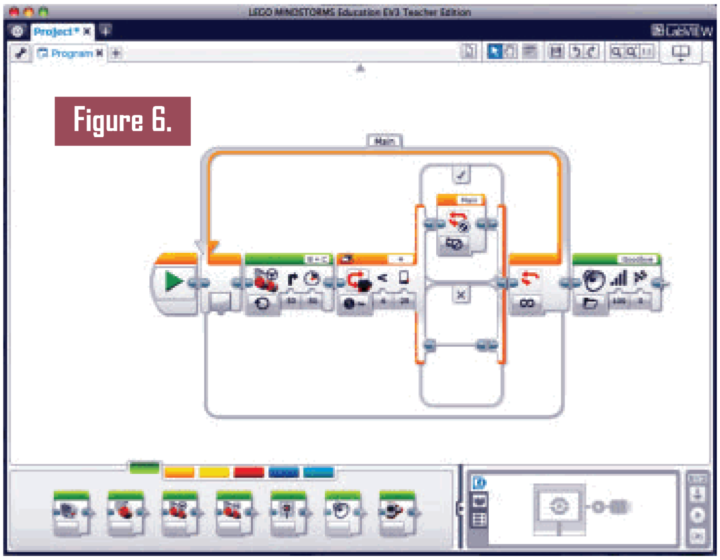
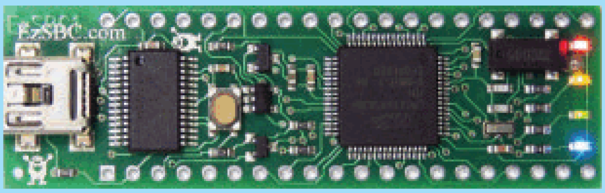


Figure 6.

I'd love to stay and write some more, but I want to see if I can make a program that will move the puppy and cat into the same region as the lady before the tidal wave hits. What was that rule about debris on the airstrip again?

If you have any questions that you would like to have answered, please drop me a line at roboto@servomagazine.com and I'll do my best to answer them.

Until next month, keep those robots running! **SV**



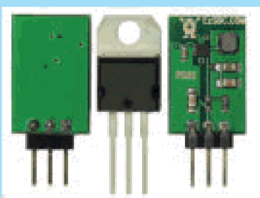
Imagine It, Control It

with our 60MHz, 32-bit controller programmed in BASIC with full 64-bit floating point arithmetic.

32 IO Pins	PWM, SERVO
I2C, SPI, UART	DHT11 Input
PULSOUT	PULSIN
Real Time Clock	16x 10-Bit ADC

Only $49.95

3-Pin Switching Regulator
Delivers 5V or 3.3V at 1A
No Heatsink required!
Only $6.95



EzSBC
Controlling your World the Easy Way

Visit us today @ www.EZSBC.com

AP CIRCUITS
PCB Fabrication Since 1984

As low as...
$9.95
each!

Two Boards
Two Layers
Two Masks
One Legend

Unmasked boards ship next day!

www.apcircuits.com

VISA MasterCard PayPal IPC MEMBER

GENUS: FROGUS ROBOTICUS

DESIGN MACHINES THAT TAKE ON A LIFE OF THEIR OWN.

The evolving field of robotics is growing fast. It requires creative and innovative thinkers with a sound education in cutting edge advancing technology.

Position your career to take off with a degree in Robotics and Embedded Systems from UAT.

University of Advancing Technology
UAT
Learn. Experience. Innovate.

uat.edu/robotics

Advancing Computer Science > **Artificial Life Programming** > Digital Media > Digital Video > Enterprise Software Development > Game Art and Animation > Game Design
Game Programming > Human-Computer Interaction > Network Engineering > Network Security > **Open Source Technologies** > **Robotics and Embedded Systems** > Serious Game and Simulation
Strategic Technology Development > Technology Forensics > Technology Product Design > Technology Studies > Virtual Modeling and Design > Web and Social Media Technologies

Please see www.uat.edu/fastfacts for the latest information about degree program performance, placement and costs.

EVENTS

ROBOTS.NET

Know of any robot competitions I've missed? Is your local school or robot group planning a contest? Send an email to steve@ncc.com and tell me about it. Be sure to include the date and location of your contest. If you have a website with contest info, send along the URL as well, so we can tell everyone else about it.

For last-minute updates and changes, you can always find the most recent version of the Robot Competition FAQ at Robots.net: **http://robots.net/rcfaq.html**.

— R. Steven Rainwater

NOVEMBER

2 **Bloomington VEX Tournament**
Bloomington, IN
Events for autonomous and remote-control robots.
https://sites.google.com/site/bloomingtonroboticsclub/

8-9 **Texas BEST Competition**
Curtis Culwell Center, Garland, TX
Remote-control robots built by student teams face off in an annual contest.
www.bestinc.org

9 **STHLM Robotic Championship**
Stockholm, Sweden
Autonomous robots compete in events that include Sumo, folk-race, line following, and freestyle.
www.robotchampion.se

10 **International Micro Robot Maze Contest**
Nagoya University, Japan
One cm cube robots compete in Micro Robot Racer and the Climbing Competition; one inch cube robots compete in Maze Solver; and there's even a two-legged robot event for tiny two inch biped robots.
http://imd.eng.kagawa-u.ac.jp/maze

16 **Atlanta Hobby Robot Club Robot Rally**
Atlanta, GA
See the website for rules and events planned for this year's Robot Rally.
www.botlanta.org

16-17 **Robotex**
Tallinn, Estonia
Events include 3 kg Sumo, iRobot Sumo, LEGO Sumo, Mini Sumo, line following, soccer, and robot racing.
www.robotex.ee

22-24 **All Japan MicroMouse Contest**
Nagareyama City, Chiba, Japan
Classic and half-size autonomous micromouse maze solving plus Robotrace.
www.ntf.or.jp/mouse

24 **Robocon**
Tokyo, Japan
Student teams from over 60 schools compete in the annual robot competition that has been held every year since 1988.
www.official-robocon.com

DECEMBER

16-19 **IROC International Robot Olympiad**
Denver, CO
Events at this year's Olympiad will focus on agricultural robots.
www.iroc.org

Be a part of the Institute of Navigation's

Annual Autonomous Snowplow Competition

January 23 - 26, 2014 Saint Paul, Minnesota

A competition devoted to the art of navigation

Be a spectator, a sponsor or a competitor!

To learn more visit: www.autosnowplow.com or www.ion.org

Sponsored by:

THE ION® AUTONOMOUS SNOWPLOW COMPETITION

Honeywell

LOCKHEED MARTIN

ASTER LABS

ATK

TORO

SPACEX

usbank

proto labs

Follow Us On:

@plowbot

Autonomous Snowplow Competition

Auto Snowplow

NEW PRODUCTS

Phone Mount

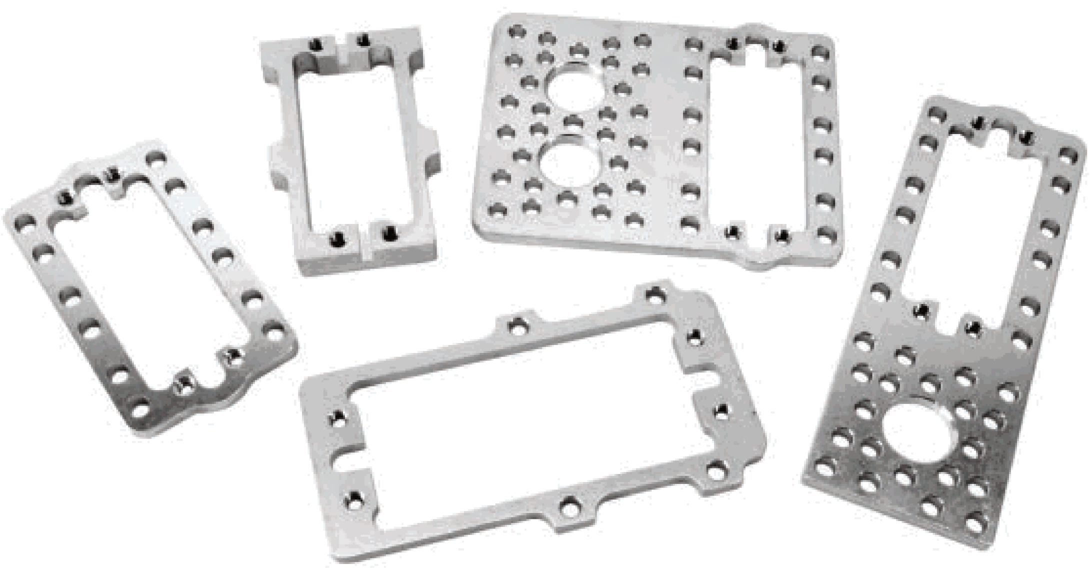
The new Actobotics™ Phone Mount A makes for a quick and clean way to attach a phone to a robotic project. Whether you're using a phone for videos, pictures, GPS coordinates, or another type of sensor feedback, this phone mount will provide a very solid and safe attachment for your device.

The Actobotics 1.5" hub pattern is incorporated into the base of the phone mount to allow for attachment to any Actobotics part that utilizes this pattern. The ball head swivel offers unlimited adjustability to provide the perfect shooting angle. The ball head is removable for users who wish to attach the camera mount to a tripod or device using a 1/4-20 camera screw. The adjustable arms of the mount have rubberized pads that will hold a phone securely and prevent scratching. It is compatible with devices with a height between 1.85" and 2.4". Price is $9.99 each.

Mounting Plates

Adding to their line of servo accessories, ServoCity recently launched five new Actobotics servo mounting plates. Four of the new plates are designed for use with standard size Hitec or Futaba servos, and a larger plate was also added to the line-up for use with Hitec's HS-785 quarter-scale servos.

All of the plates are machined from 6061-T6 aluminum for superior strength and durability. The Actobotics channel pattern can be found on two of the mounts, allowing for easy attachment to other Actobotics components.



The other mounts also incorporate mounting holes that are compatible with the Actobotics product line.

Simply insert a servo, and tighten the plate to the four servo mount tabs using 6-32 screws. Prices start at $4.99 each.

For further information, please contact:

ServoCity	Website: **www.servocity.com**

C-Programmable Robot Kit

Global Specialties has introduced their new ASURO robot kit. The ASURO is a small autonomous multi-sensored robot developed for educational purposes by the DLR which is the German Aerospace Center (like NASA).

Highly versatile, the ASURO is completely programmable in C. Assembly is easy for experienced electronic technicians and feasible for a novice. Except for the printed circuit boards (PCBs), only standard parts are utilized and freeware tools can be used for programming.

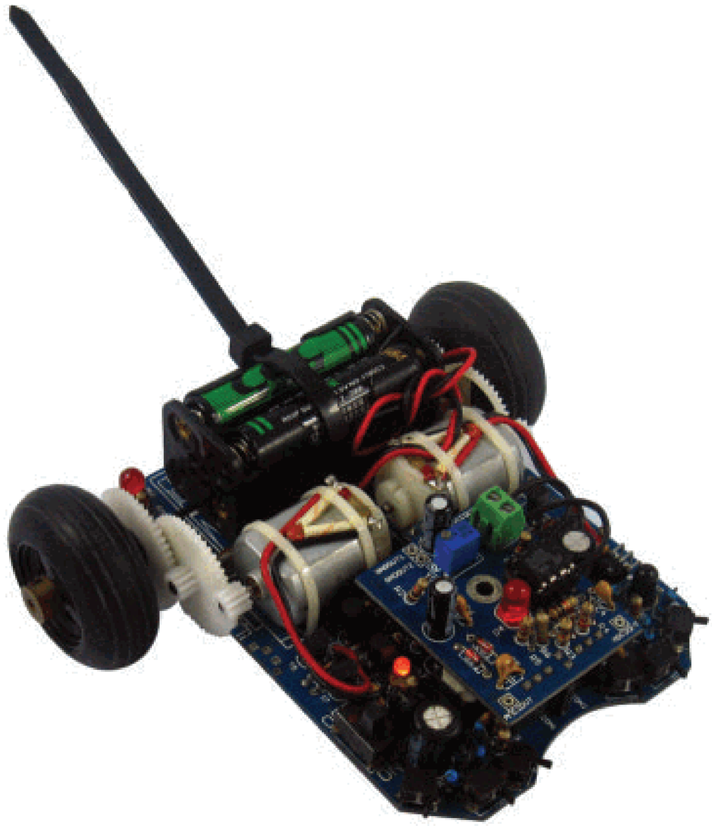
The ASURO comes unassembled and includes a soldering guide, making it suitable as an introduction into processor-controlled hobby electronics for projects in schools, universities, and technical education.

Features include: an ATmega8L, eight-bit AVR-RISC processor; it's fully programmable in C language; it comes unassembled so soldering is required; it has a CD with software, training manual, and supporting materials; and there is AVR-GCC freeware for use with Windows or Linux.

ASURO also includes a unique and safe USB IR transceiver with simple to operate Flash software; remote control and possible PC-programming via a USB transceiver; possible wireless control possible with optional Bluetooth and 433 MHz RF; extensions kits are available; it has six collision-detector sensors; an optical line-tracker unit; two independently controlled 3 VDC motors; an onboard odometer sensor on both wheels; preprogrammed firmware for hardware testing; three LED indicators; and a one year warranty. List Price is $ 99.

For further information, please contact:

Global Specialties Website: **www.globalspecialties.com**



Dynamixel Pro Line-Up of Developer Products

ROBOTIS — manufacturers of the Dynamixel smart servo that is widely used for robotics education and research — has released Dynamixel Pro: a new line-up of products targeted for professional robot developers in the field/service robotics market.

The basic all-in-one modular concept of Dynamixel has been inherited by the Dynamixel Pro. It has integrated the BLDC motor (up to 200W), controller, driver, encoder, and gear system into its compact rectangular shape. The key innovation has been made to the cycloidal gear system which guarantees minimum backlash as low as 3.5 arcmin, yet maintains endurable operation.

In addition to feedback information like position and speed, Dynamixel Pro has a current-sensing feature that enables force control algorithms for service robots.

Features include a 32-bit ARM cortex embedded incremental encoder; plus, an absolute magnetic encoder;

Continued on page 74

INVASION OF THE JELLYFISH

In South Korea, jellyfish are threatening marine ecosystems and are responsible for about US$300 million in damage and losses to fisheries, seaside power plants, and other ocean infrastructure. Large jellyfish swarms have been drastically increasing over the past decades and have become a problem in many parts of the world. Hyun Myung, a robotics professor at the Korean Advanced Institute of Science and Technology (KAIST), said in a recent interview they aren't just affecting marine life and infrastructure any more. "The number of beachgoers who have been stung by poisonous jellyfish — which can lead to death in extreme cases — has risen," he says. "One child died due to this last year in Korea."

So, Professor Myung and his group at KAIST set out to develop a robot to deal with this issue, and have tested out their solution: the Jellyfish Elimination Robotic Swarm (JEROS) in Masan Bay on the southern coast of South Korea. They've built three prototypes like the one shown here.

The JEROS robots are autonomous and are able to use cameras to locate jellyfish near the surface, Professor Myung explained.

Due to the large number of jellyfish, developing some sort of catch-and-release mechanism is just not feasible, so the robots are equipped with hardware that would probably be considered inhumane to use on anything with a backbone.

Together, the JEROS robots can mulch approximately 900 kilograms of jellyfish per hour. Your typical moon jelly might weigh about 150 grams. So, that's about 6,000 ex-jellyfish per hour.

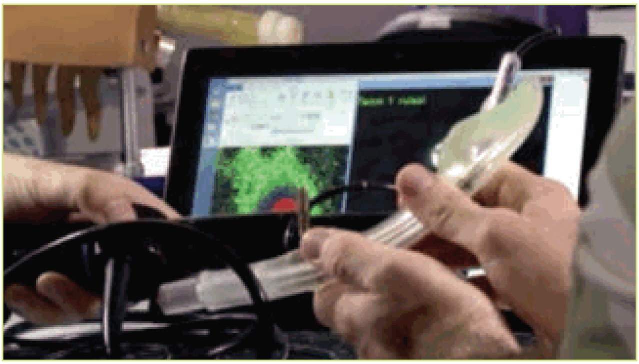
Professor Myung says that because the robots are designed to work cooperatively, adding more units to deal with the large quantities of jellyfish shouldn't be a problem. His team is already planning more tests in their efforts to deter the gelatinous invaders.

BREATHE EASIER

A robotic device that crawls into lungs could help deliver vital air to patients, researchers say.

To help anesthetized or critically ill patients breathe, flexible plastic tubes are placed into the lungs to maintain an open airway — a procedure known as intubation. Currently, intubation requires physicians to look down the throat and choose between two very similar openings: one leading to the lungs; the other to the stomach.

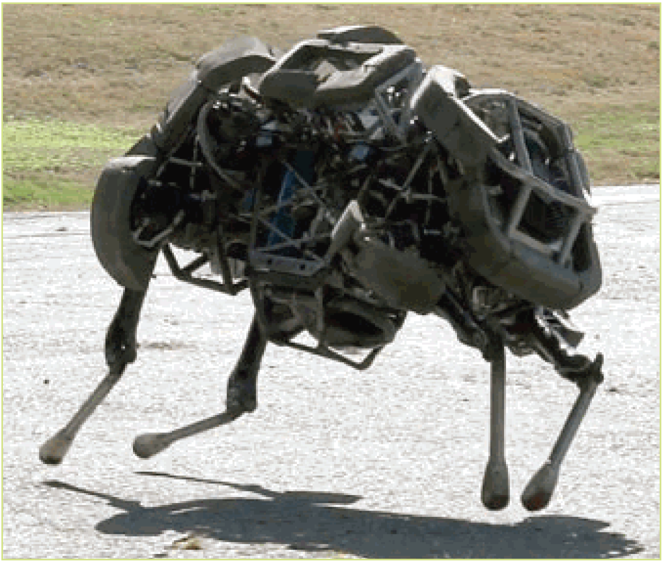
Picking the wrong opening to intubate can lead to death. Moreover, intubation sometimes has to be performed in challenging situations that can make the procedure even more difficult, such as the battlefield or with fluids like blood obstructing the way.



A new robotic device aims to improve intubation procedures, especially those done in challenging situations.
(Photo courtesy of the Hebrew University of Jerusalem.)

Now, scientists have revealed a robotic intubation device that can automatically identify the lungs.

A prototype of the device — called the GuideIN Tube — was successfully tested on cadavers at the Hadassah Medical Center in Jerusalem.



GREAT GALLOPING GAITS!

Boston Dynamics recently unveiled their newest creation: WildCat — a totally new quadruped robot based on Cheetah that untethered reaches speeds up to 25 km/h (16 mph).

WildCat is a four-legged robot being developed to run fast on all types of terrain. So far, WildCat has run at about 16 mph on flat terrain using bounding and galloping gaits. WildCat is being developed by Boston Dynamics with funding from DARPA's M3 program.

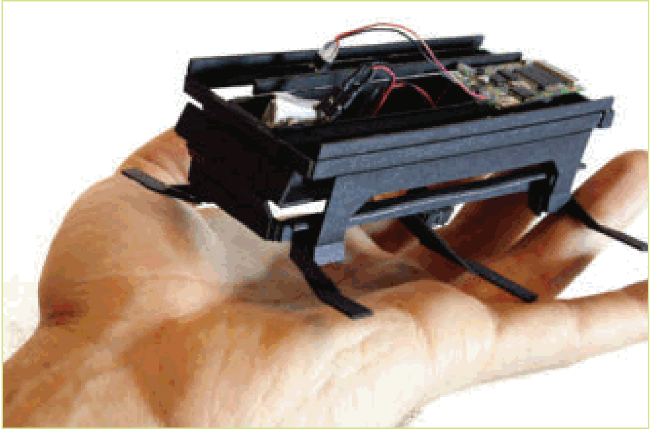
SNAKES ON A PLANET

SINTEF ICT (Applied Cybernetics), CIRiS (Centre for Interdisciplinary Research in Space), and NSC (the Norwegian Space Centre) are cooperating in the SERPEX project which aims at investigating the feasibility of employing snake robots in future planetary exploration missions. The project is funded by the ESA (European Space Agency).

Researchers at SINTEF and CIRiS see great potential in the use of snake robots in space missions involving planetary exploration. A snake robot can, for instance, work together with a planetary rover through a tethered connection, and can also act as the manipulator arm of the rover when it is not crawling freely. The SERPEX project will identify advantages, disadvantages, possibilities, and challenges related to the use of such mechanisms in a space mission context.

The long-term motivation behind the project is the development of a robotic propulsion mechanism that can reach and operate in locations not accessible by existing planetary rovers. The many application areas of snake robots on earth imply that the technological development of a snake robot for space missions will have strong synergies with related earthbound applications. Many industries and application areas on earth can both support and make use of the technological elements of a snake robot developed for space missions.

Find more details at **http://robotnor.no/research/serpentine-robots-for-planetary-exploration-serpex/#sthash.JAluBhj1.dpuf**.



DASHING ROBOTS

Dash Robotics' bio-inspired bots have been one of those research platforms that lives in a laboratory and rarely gets to come out and play. Until now, that is.

Dash has been turned from a research platform into a robot that you can buy. Even more remarkable is it's actually affordable. A beta version that includes a steerable robot (that you assemble yourself in an hour with a little bit of glue), along with a full electronics package is a mere $65. You can get it in blue, orange, yellow, or black.

The robot is controlled with Bluetooth via a mobile device, and a variety of onboard sensors will enable a range of apps from obstacle avoidance to "photovore" behaviors.

Dash creators are hoping to enable folks to make clever things with their bots:

"We are developing a custom electronics package that is Arduino-compatible, uses Bluetooth 4 communication, has a dual motor driver, several LEDs, and connects with Micro USB. The battery can be charged through the Micro USB connector and lasts about 40 minutes. The electronics are plug-and-play, so you can run Dash without any programming knowledge. But we've also made him hackable, so you can take advantage of the sensors we've included, or even add your own."

Out of the box, builders will have access to:

- Gyroscope
- Visible light sensor
- LEDs (red, green, yellow)
- IR emitters and sensors
- I/O pins for expandability
- Bluetooth low energy communications
- Micro USB connector

The Beta Dash (which is probably the one you want) isn't called Beta just for fun. By participating in this crowd-funding campaign, you'll be helping to figure out what the next generation of Dash will be like. It's probably safe to say that the next generation will be a lot bigger than this first one, which is limited to just a thousand robots.

Expect the Beta Dash kits to ship in early 2014. Of course, they may have already sold out.

SCOUT EARNING ITS BADGE

Scout — a four meter long autonomous boat built by a group of young DIYers — is attempting to cross the Atlantic Ocean. It is traveling from Rhode Island, where it launched on August 24th to Spain, where (if all goes well) it will arrive in a few months' time. Scout has now gone about 1,000 miles (1,600 kilometers) of its planned 3,700 mile (5,900 kilometer) journey. Should it complete this voyage successfully, its passage will arguably belong in the history books. Although the construction of Scout's hull is somewhat high tech — carbon fiber sandwiching Divinycell foam — the rest of the boat is comparatively simple. Solar panels mounted on the top of the hull charge a lithium-iron-phosphate battery which, in turn, powers an ordinary trolling motor attached to the bottom of the hull.

EDMUND
SCIENTIFICS®
Wonder and Inspiration Delivered — Since 1942

1-800-818-4955 • ScientificsOnline.com

$5 GROUND SHIPPING
for Servo Subscribers
USE **SSCTE3**
AT CHECKOUT*

The Brightest Comet of Our Lifetime Arrives This Fall

All eyes this autumn will be on ISON, the potentially record-breaking comet that has astronomers the world over predicting it might be the "Comet of the Century." Coming within 800,000 miles of the Sun's surface in late November, this 'sungrazing' comet should shine as bright as Polaris–or brighter!

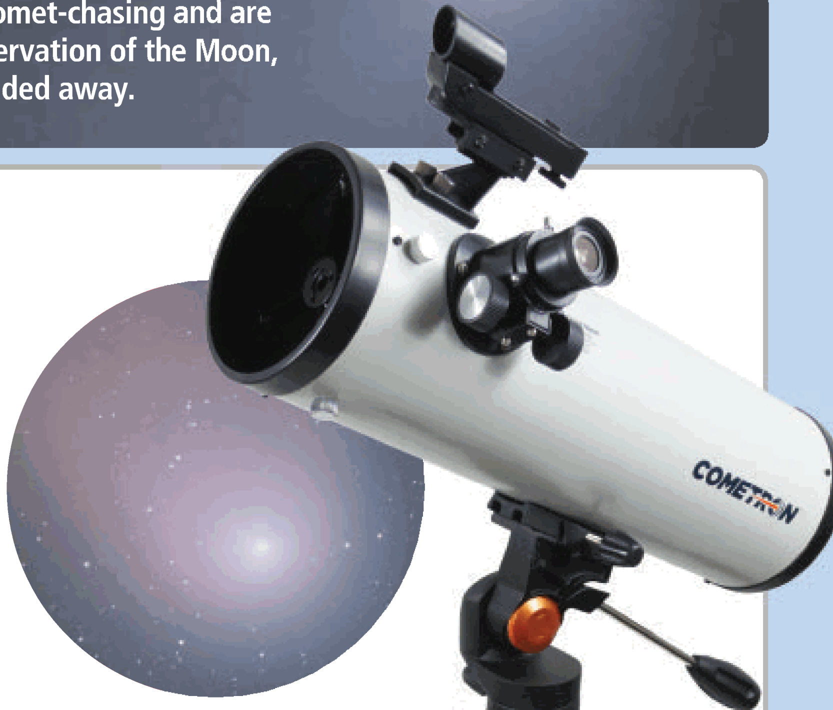
These selected Celestron products are ideal for comet-chasing and are guaranteed to maximize your excitement for observation of the Moon, stars, clusters, and nebulae long after ISON has faded away.

Stargaze Anywhere with a Large-Aperture Reflector

A large-aperture reflector design provides a wide field of view, ideal for observing bright comets, galaxies and star clusters. Features a parabolic mirror for the sharpest image and easy, no-tool setup.

- Newtonian Reflector telescope with 114mm (4.49") aperture
- Magnification: 16x-269x
- Finderscope: StarPointer Red Dot Finder
- Adjustable stainless steel tripod
- Eyepieces: 20mm (22x) and 10mm (45x)
- Alt-azimuth mount with panning handle

Celestron 21079 Cometron 114AZ Telescope
#3153327.....$179.95



Simple Design Makes it Easy for Beginners to Use

Upgraded for 2013, the Cometron FirstScope is a compact, tabletop telescope perfect for beginners looking to experience the magic of a dazzling comet, planet or nebula.

- Dobsonian telescope with 76mm (3") aperture
- Magnification: 15x-30x
- Eyepieces: 10mm and 20mm Kellner

FirstScope 76 Telescope
#3153326.....$59.95

Track Comets and Find Stars Instantly with This Powerful Binocular

This binocular's massive 70mm objective lenses, multi-coated optics and BaK-4 prisms provide bright, sharp images of comets and celestial objects. Designed for use with your tripod (not included).

- Objective lens diameter: 70mm
- Magnification: 12x
- Includes lens covers, neck strap, tripod adapter & carrying case

Cometron 12 x 70 Binocular
#3153329.....$89.95

FIND THOUSANDS OF OTHER UNIQUE SCIENTIFIC ACTIVITIES, GADGETS, COLLECTIBLES AND MORE.
SCIENTIFICSONLINE.COM

$5 ground shipping valid on featured products only through 12/15/13

COMBAT ZONE

Featured This Month:

Go to **www.servomagazine.com/index.php?/magazine/article/november2013_CombatZone** for any additional files and/or downloads associated with this article. You can also post comments.

SPECIAL REPORT:

Judge Dave's Guide to Winning

*Editor's Note: Having competed in this sport for four years and reported for eight more, you'd think I would have run across this gem before. I guess it goes to show that there's a lot of stuff on that there Internet. The following article is reprinted from the ComBots website (**http://combots.net/guide-to-winning.php**), courtesy of author Dave Calkins.*

Judge Dave (Calkins) is one of the original BattleBots judges, founding RFL member, founder of RoboGames, and co-founder of ComBots. These are his words of wisdom. Ignore them at your peril.

Since the year 2000, I have judged over 2,500 robot combat matches, and watched close to 5,000 (a given event can have anywhere from 100-500 matches — they add up after 10 years). All of the following rules come from watching contestants cry at the smell of burnt metal and broken dreams. Follow these rules and you'll do well. If you don't, you won't.

1. Know Carlo's Law and live by it.

Carlo Bertocchini — builder of BioHazard (three time heavyweight champion) — came up with this most important rule. He put it better than I can, so here it is:

Finish your robot before you come to the competition!

This seems too obvious to even mention, let alone to place at the very top of a list of secrets to success. Besides, so what if you just have a little wiring to do, or that one last gear to mount? It's 3:00 AM, you have been working for 36 hours straight. You can do that last bit of wiring in the pits, right? Well, the fact is if you are in this situation, you have probably already ensured a loss in the arena.

"Moe" got to the event with just a few "minor" adjustments left to do. He spent his whole day trying to work on the robot while at the same time getting through all the required safety check-ins. He was somehow able to convince the inspectors that his robot was safe and able to move under its own power.

Now it is the first day of competition. Moe is still working on his robot after having slept just two hours last night under the pit canopy. Moe found that the minor adjustments took longer than he expected, and he found a few more changes that just had to be made.

Now Moe is called to battle. He sets up his robot and steps out of the arena. The box is locked. The blue driver is ready.

"Red driver are you ready?"

"Uh, I guess."

Three thousand people watch anxiously from the stands as the starting lights count down to green ... Moe's bot never moves. Three thousand people watch with disappointment and ill-disguised hatred as Moe walks into the box to collect his robot.

What I am suggesting here is not easy. It takes good planning, discipline, and lots of free time to get the job done. However, here is one simple way to guarantee that your robot will be finished: If it looks like time is running short, rather than drive or fly hundreds of miles just to work on your robot at the venue, why don't you just leave your robot home! Come and see the show, have the time of your life, learn a few things, and set your sights on doing well and enjoying the next competition.

In other words, if it isn't fully functional the week before the event, it's probably not going to pass safety or be able to fight. It's happened countless times. Save yourself the shame. Come to the event no matter what — but if your bot's not ready, volunteer on another team.

2. Practice driving (LTFD).

Sounds obvious, I know. So do all the other guidelines, but less that 20% of contestants obey this rule. The ones who do are the ones who win. So many competitors spend countless hours making tiny little changes to their robot to make it "perfect" that they don't spend any time driving it. A shocking number of rookies have had no driving practice before they step into the arena for their first match.

Listen guys, Dale Earnhardt Jr. didn't just hop into the driver's seat and start winning at NASCAR. Pay very close attention to this next sentence because if you want to win at RoboGames, ComBots Cup, or any other competition, it's the most important thing I can say to you: The single greatest common denominator to winning is driving ability. Get that?

The first time I saw Gary Gin (three-time HW champion) and Original Sin and Big Bee, I thought it was a joke. His bots had no weapons. Exposed wheels. Soft aluminum bodies. However, he drove circles around his competitors. He deftly avoids spinning blades, flippers, and whatever else is thrown at him. He strikes and dodges — like the finest boxer. He wins time and time again. All the greats are like that. Watching Matt Maxham (four-time champion) drive Sewer Snake is like poetry in motion. They win because they practice driving their robots! It doesn't matter how great your weapon is if you can't actually hit the other bot.

Spend 100 hours practicing driving before you ever get to the event. Robot not done yet? Fine. Go spend $20 on a cheap R/C car and drive until your robot's ready. Switch to your bot as soon as the drive train is finished — even if the weapon isn't done and the armor isn't on. Spend an hour each day driving. Go find some empty lot, parking garage, or cul-de-sac. Now, chase that $20 R/C car around with your bot (let the kid next door drive the car; he's probably a better driver than you, anyway). Make sure you can catch it. Corner it. Out-maneuver it. Dominate it. When you compete, the guys you fight against are moving. Practicing against an unmoving target is worthless.

Got that down pat? Good. Now, disconnect a motor. Learn to drive with any given motor disabled (I've seen Gary control his bot with only one of four wheels left. This came from practice, not magic.) All of these things will happen in the arena, and you can either learn now, or learn then. Your choice.

(Editor again. Sorry to butt in, but I just have to give a shout out to the late Steve Judd. If I saw him post this once on forums, I saw it 678 times. Every new guy who asked, "What do I do before an event?" got the same answer: "LTFD.")

3. Be able to self-right.

It is not a question of *if* your robot will be flipped over, it is only a question of *when* your robot will be flipped over. I have seen competitors, their eyes filled with tears as they take their magnificently engineered robot out of the arena after a loss, saying "I was so sure we wouldn't get flipped."

Wrestlers get body slammed. Quarterbacks get dog piled. Skiers dump skis along a quarter mile path. What makes you so sure you won't get flipped? I've seen countless matches where Robot A was utterly dominating Robot B and would have won by a landslide if it were a judge's decision. But then, by bad luck, bad driving, or just a big collision, BAM! Robot A is upside down and loses the match.

Your robot must be able to either self-right (flipper, actuating arm, whatever) or operate upside down (wheels extend above and below the robot).

If you can't self-right, you'll never make it to the finals. Count on that. If there is any position in which your robot is a helpless kitten, count on it ending up that

way at some point during the competition.

4. Simulate getting attacked.

Okay, so you've finished your bot with a few months to spare. This is the piece of advice that you are just not going to want to take.

I want you to go to the hardware store. Buy the biggest sledgehammer you can find (the really big kind that makes you strain when you lift them). Now, raise it above your robot ... and beat the living hell out of it.

Awe, did it bweak? Issums widdle wobot in a big pile uv parts??? Well, I just saved you the indignity of having that happen while 3,000 people watched. If your robot cannot survive a good bashing with a sledgehammer, circular saw, or a 10 foot free-fall, it will not last in the arena.

Use good 6061 or 7075 aluminum, steel, or (preferably) titanium. Make sure you have a good enough infrastructure to support your outer shell.

Ensure that all components are securely mounted. They're going to get knocked around. I lost count long ago of the number of battery packs that I have seen flying across the arena because they got knocked out of the bot in a big hit. If you lose your batteries, you lose the match. It's that simple.

Drop your bot off the roof of your garage. No, really. It's a good simulation of what's going to happen when a bot like Ziggy flips it 15 feet in the air, or a spin-bot like Last Rites whacks it once and sends it flying across the arena. Your bot has to be able to withstand that kind of hit. Even if you're the best driver in the world, you're still going to take lots of knocks (including on the bottom of your bot, so have undercarriage armor as well). You must be able to survive those hits, and your first match is the wrong time to find out where your weak spots are.

(Sigh. Sorry. Editor again. Use the TeamPyramid approach. Put your machine in the back of a pickup, hit 20 mph in an abandoned concrete parking lot, and drive the bot off the back of the truck. Then, chase it around while the passenger drives the bot. Sort of like a game of chicken.)

5. Have a weapon system.

Better yet, have two. This is robot combat. You don't play baseball without a bat; you don't go to war without a gun; and you don't become a pro-wrestler without having at least two frontal lobotomies. If you want to beat the daylights out of the other robot, bring a weapon!

Wedges can be effective, but it's extremely rare for a wedge with no other weaponry to make it to the finals. Watch lots of matches (everything you can from YouTube and other videos, or better still, buy a three-day pass for an event) and take lots of notes. See what weapons work and which don't. Think about *why* things worked. Two weapon systems that look identical may operate completely differently, with very different results.

Better yet, come up with a new and unique weapon system. Something that hasn't been tried before. Every time I go to a competition, somebody has brought along a new robot which garnishes lots of oohs and ahs from the masses — and more than a few times of "Why didn't I think of that?" Just make sure the weapon is allowable in the rules (no liquids and no tasers).

6. Simulate attacking.

I swear some people show up to a competition having only ever tested their robots on kittens. Sure, it may give your garage a nice new primer coat of kitty juice, but that doesn't mean it will even scratch the paint on another 1/4" steel armored robot.

I walk the pits before competitions and between matches to see who's doing what and how this year's robots are sizing up. During one event, I saw a well designed super heavyweight with a horizontal spinning mass (that's our technical term for a big spinning hunk of metal). Except the metal bar had not a single ding on it. You can give something a nice coat of paint, but you can't hide the dings. No scratches. Nothing. On closer inspection, I noticed that the bar (which probably weighed 40 pounds) was held to the rotating shaft with a half ounce cotter pin. The kind your six year old niece can bend with her pinky.

"You guys test this against anything?"

"Of course not, it could hurt someone!"

The first time that metal bar hit another robot, the pin sheared, the bar went flying, and they were done. If they had spent five minutes in their garage or at some junkyard testing their weapon against a solid object, they would have realized the cotter pin was a weak link and they could have fixed it.

There's a term for this: Cargo Cult. It comes from South Pacific islanders who got used to planes coming in during World War II bringing supplies. After the war, the planes stopped coming. So, islanders fashioned headsets from coconuts, built runway towers, and made landing lights. Still, the planes never came. Just because something looks the same, doesn't mean it will work the same. Don't be a cargo cult competitor.

While you're testing, make sure that you're able to actually push twice the amount of dead weight as the maximum in your weight class. This will be a good simulation of a bot pushing against you. If you can't push that much weight, you're probably going to lose.

A great many matches come down to pushing matches (fifth round, both bot's weapon systems are out, half the armor is gone, and there's a burned out speed controller), so you need to be sure you can win under these circumstances. It's also another time to find out if your speed controller can handle the load,

or if it's going to give up its magic smoke.

7. Go to a competition, watch as many matches as you can, and take notes.

If you're a contestant and you've lost, don't go home and sulk. Go sit in the stands and watch every damned match until the finals are done. I've seen too many cry-baby first timers leave immediately after their first loss. (Michael Jordan got cut from his high school basketball team — he did NOT go home and sulk.) You can learn more from other people's victories and mistakes than just your own. So, sit back, relax, and enjoy the show.

Be sure to take notes. Your memory's not that good, trust me.

8. Use good batteries, have spares, and make sure they'll last five full minutes.

When you start building bots and playing with them, you're going to learn one lesson the hard way (you won't learn it here, trust me). Batteries get hot. REAL hot. And they take forever to recharge. At least in robot combat time. So, you should have easy access to replace your batteries between matches. Have at least two full sets (three or four if you can afford them). Have one on the charger, and one in the bot.

As soon as a match is over, put your just-used batteries on the charger. Just before a match, take the fresher pair off and install them. No matter how good a driver you are or how well built your robot is, if the batteries don't last the match, you're not gonna win.

9. Don't let the judges decide the match for you.

Matches are judged based on the full three minutes. The first minute is as important as the last. The fact that you kicked ass the last 20 seconds doesn't make up for the first 160 seconds when your competitor was mopping the floor with your rivets. You want to avoid narrow losses? Want to avoid a screaming match with the officials because they didn't share your belief that your completely out-of-control robot was actually using a strategy?

Simple. Go for a knockout. Don't let the match last all three minutes. Design your robot and operate it so that you kill the other robot so the referee counts it out. Keep your fate in your hands, don't put it in the judges. Judges are painfully fair and unbiased. The problem is you're not. You want your bot to win and the other team's bot to lose. They don't care who wins. It's just that in a close match, they make the call. Both sides think they've won, but only one of them will be correct.

If the judges think the other robot was more aggressive and did more damage, then you're going to lose. But judge's opinions don't have to matter. All you have to do is knock the other bot out. Do not hesitate. Do not unstick it. Do not try to avoid extra damage. You are there to win. There's only one way to absolutely ensure that you win: Go for the knockout. Every single match.

10. Read the damned rules.

I cannot count the number of builders who have spent hundreds of hours and thousands of dollars building their dream-come-true, and didn't spend one small hour reading the rule-book from cover to cover. You need to do this for every competition; they change from season to season.

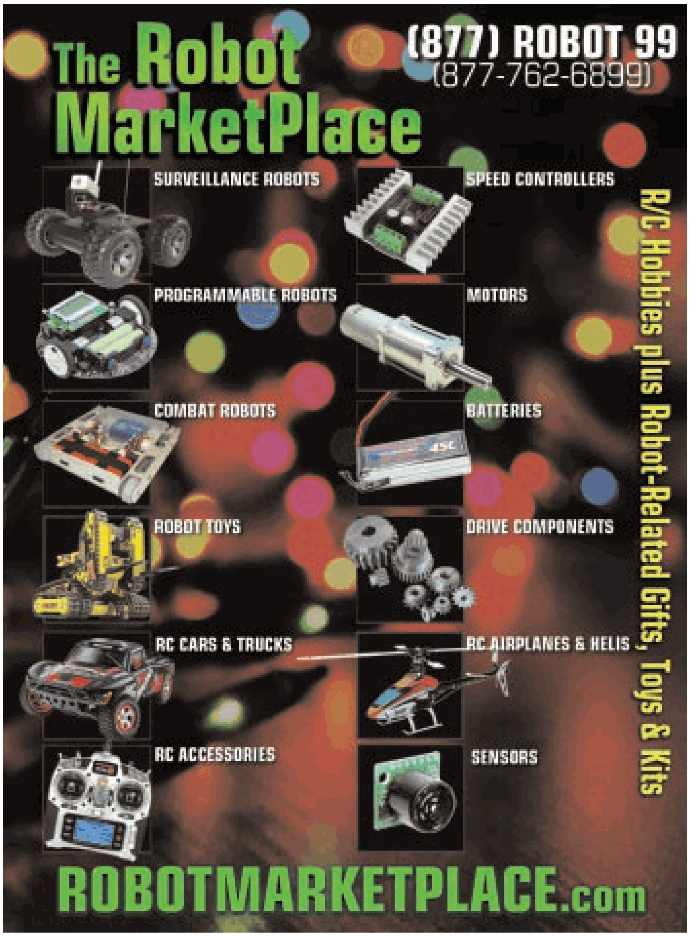
* Know exactly what the judging criteria are. Hint: Number of hits is not part of the judging criteria.

* Know what weapons are allowed and what's not.

* Understand how to pass safety (if you don't pass safety, you don't compete).

* Understand what can get you disqualified.

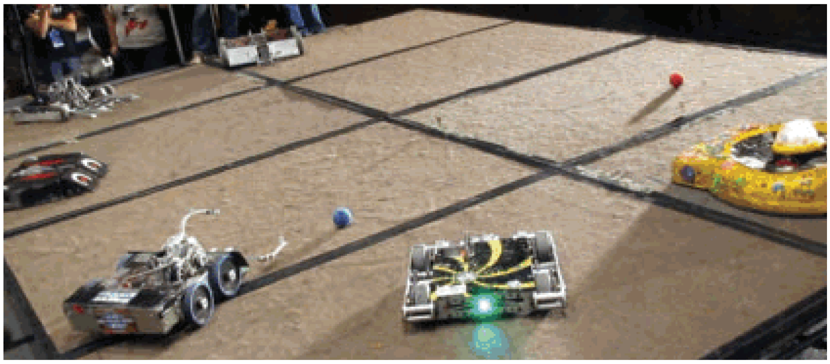
If you can't spend an hour reading the rules (don't think that you know them just because you've seen every episode on TV), you probably will never get to compete, much less win. SV



EVENT REPORT:

DragonCon Robot Battles 2013

● by Mike Jeffries



The Robot Battles combat surface before one of the 30 lb rumbles.

Robot Battles 47 took place over Labor Day weekend during DragonCon in Atlanta, GA. At DragonCon, Robot Battles consists of two independent events. On Sunday, the one and three pound machines compete in an eight foot arena; on Monday, the 12 and 30 pound machines compete in a Sumo style competition on a set of stage risers. The Robot Battles rules can be viewed at **http://robotbattles.com/**.

Eleven 12 pound bots and eight 30 pound bots competed in the Robot Battles event. As is common in these events, a range of strange and wonderful machines turned up to compete in a double elimination tournament for the title of Robot Battles champion.

Among the 12 lb entries were Oops (**Figure 1**), a strong aluminum brick; Meat Head (**Figure 2**), a powerful pusher; 1 Minute Bot (**Figure 3**), a wedged pusher and the 2012 runner up; Steel Cordon Bleu (**Figure 4**), a robot with ramming spikes and grippy wire brush wheels; Omega Force (**Figure 5**), an invertable bot with a kinetic flipper; and Served Cold (**Figure 6**), a massive 12 lb robot with a dustpan style scoop.

Among the 30 lb entries were Trident (**Figure 7**), a wooden wedge; Mister Fun (**Figure 8**), a wedge with a strong drive system that was missing its intended hammer weapon; Null Hypothesis (**Figure 9**), a fast powerful ramming wedge; and Dandelion's Revenge (**Figure 10**), the carcass of a lawnmower with the still-working popper of a child's toy mounted to the top.

In the 12 lb and 30 lb classes, each match is fought in a Sumo style format with the winner being the first bot to reach two wins. In the event both bots leave the combat surface at approximately the same time, the bout is declared a draw and the robots return to their starting locations. The lack of arena walls makes for fast-paced matches with often surprising results.

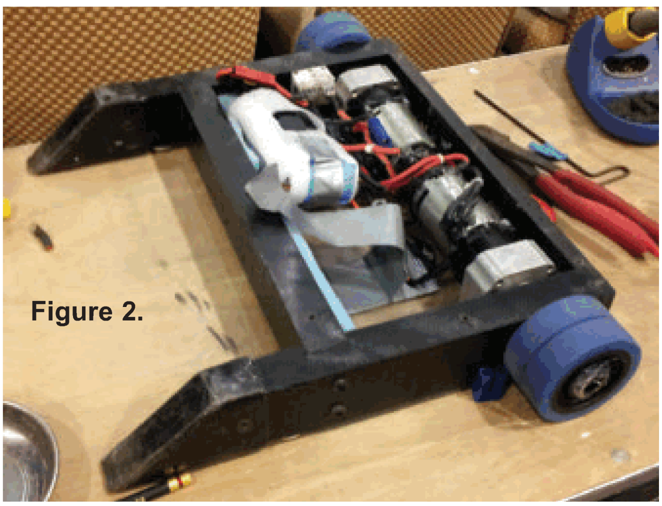


Figure 2.

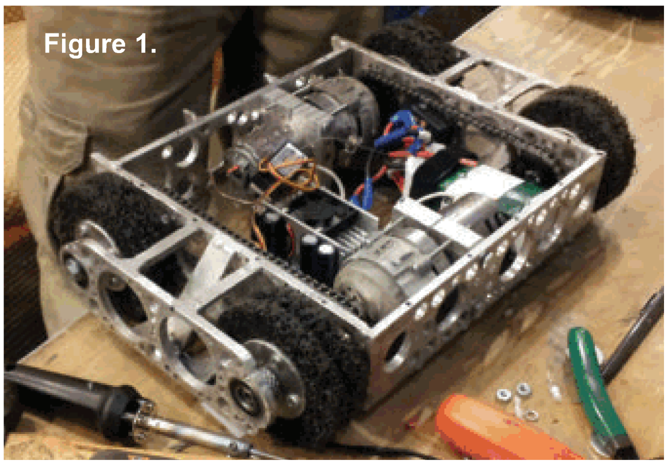


Figure 1.

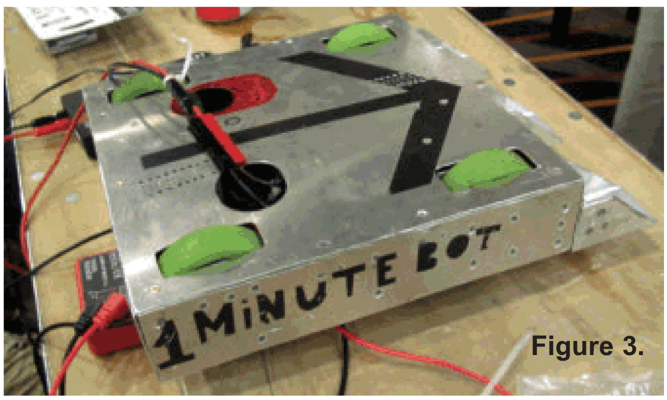


Figure 3.

Figure 4.

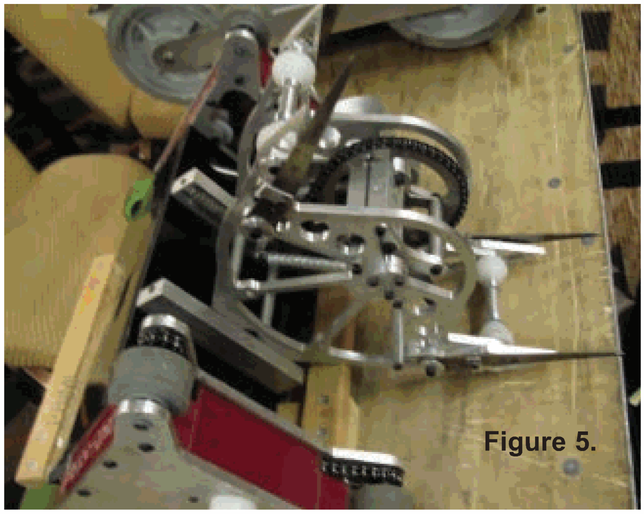


Figure 5.


Figure 6.

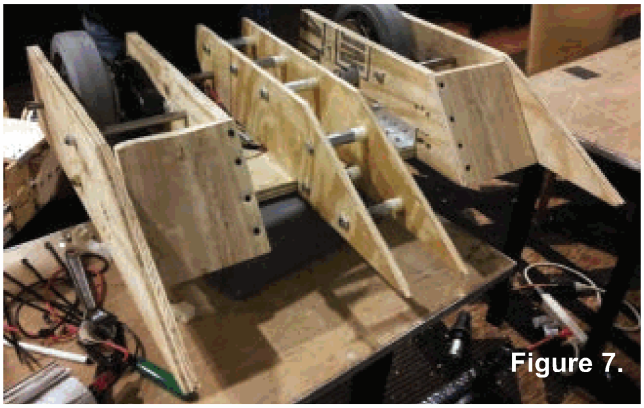


Figure 7.

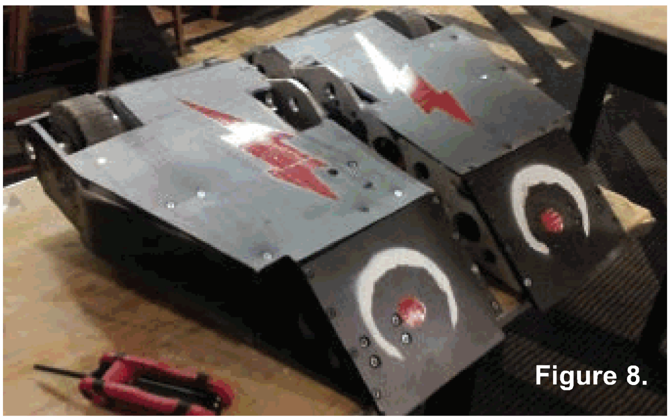


Figure 8.

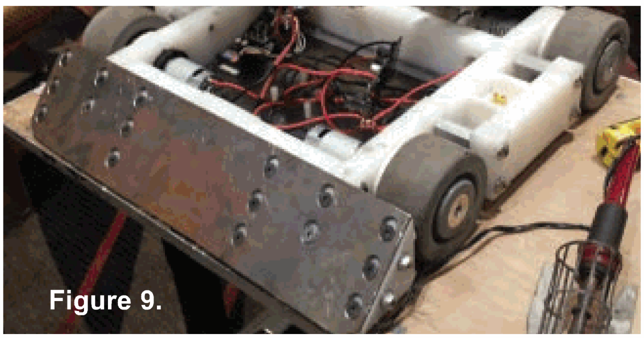


Figure 9.

The 12 lb final four were 12 O'Clocker (**Figure 11**), Apollyon (shown in **Figure 17** with Nyx), Dead Meat (**Figure 12**), and Tetanus Shot (**Figure 13**). Dead Meat and Tetanus Shot fought in the loser's bracket semi-final, with Tetanus Shot advancing to the loser's bracket final.

In the winner's bracket final, Apollyon beat 12 O'Clocker sending it to the loser's bracket final. Tetanus Shot managed to eliminate 12 O'Clocker to move on to the overall final. Apollyon beat Tetanus Shot to secure the


Figure 10.

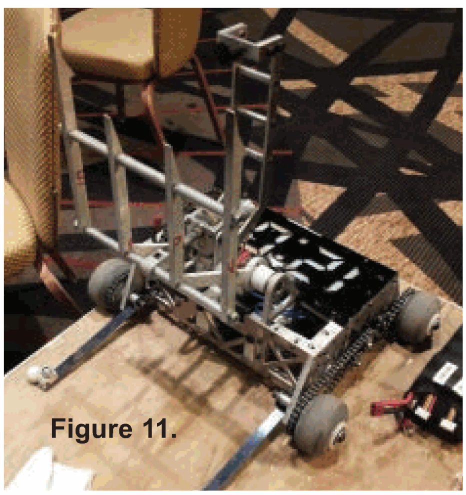


Figure 11.

win and its third consecutive 12 lb championship at DragonCon .

The 30 lb final four were Big Blue Saw Presents Jaws (**Figure 14**), Nyx (**Figure 17**), Overthruster (**Figure 15**), and Uberclocker Advanced (**Figure 16**). Uberclocker Advanced and Big Blue Saw Presents Jaws faced off in the loser's bracket semi-final with Uberclocker Advanced heading to the loser's bracket final.

In the winner's bracket final, Nyx beat Overthruster sending it into the loser's bracket final. Uberclocker Advanced avenged an early tournament loss beating Overthruster to advance to the overall final.

In the final, Nyx came out on top after a series of close matches to win its second consecutive 30 lb championship at Robot Battles.

	12lb	30lb
1st	Apollyon	Nyx
2nd	Tetanus Shot	Uberclocker Advanced
3rd	12 O'Clocker	Overthruster

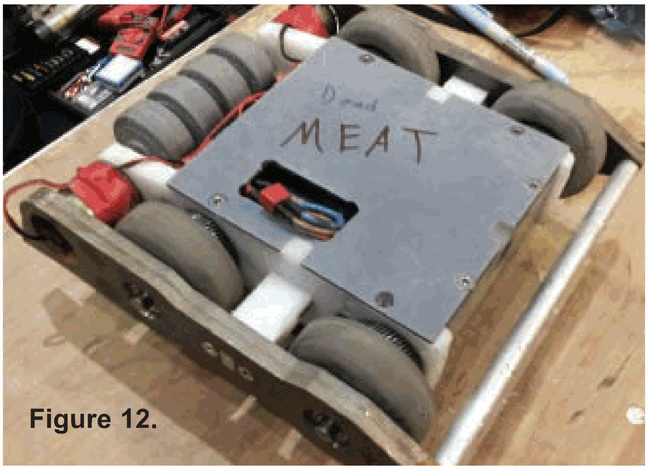


Figure 12.

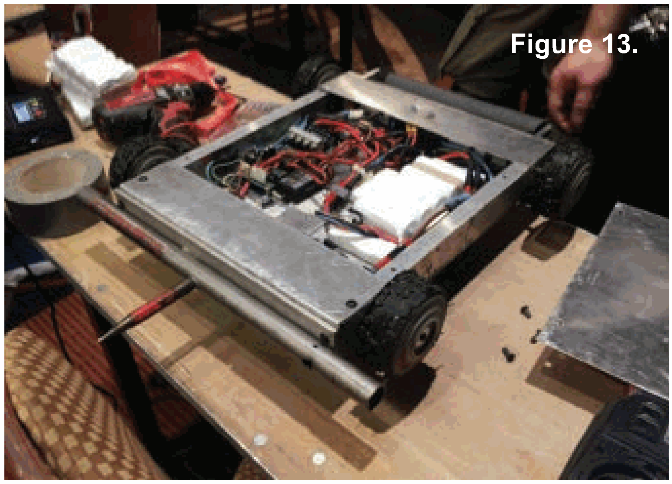


Figure 13.

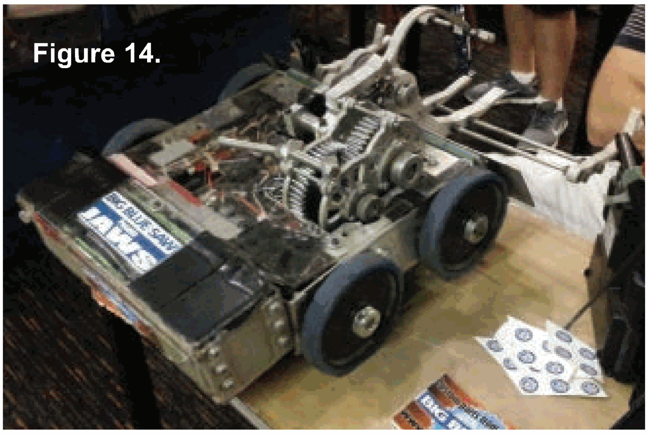


Figure 14.

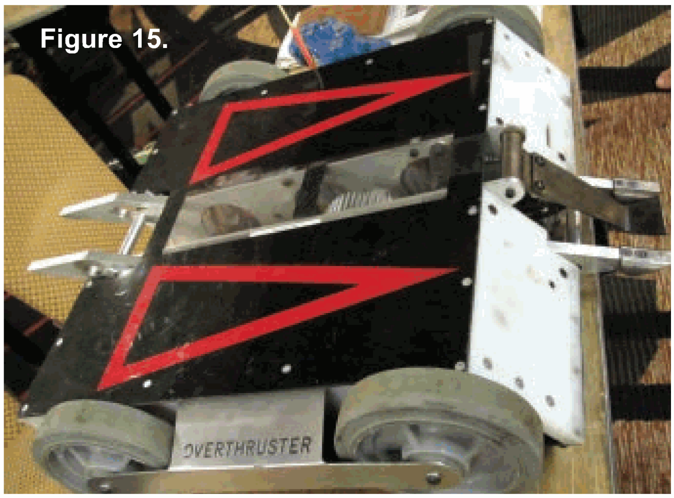


Figure 15.

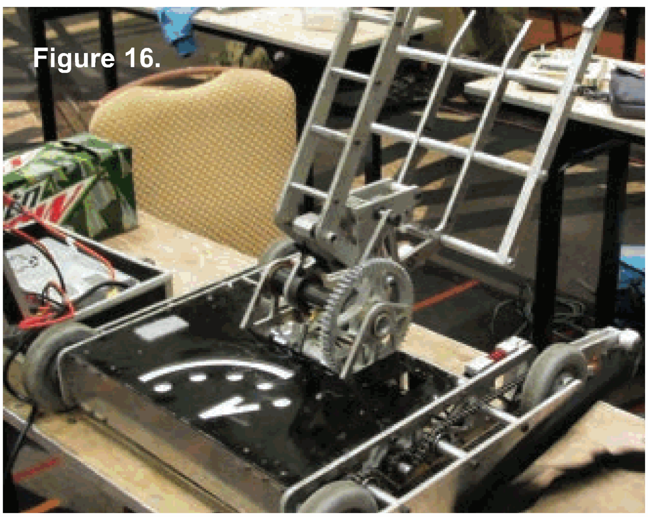


Figure 16.

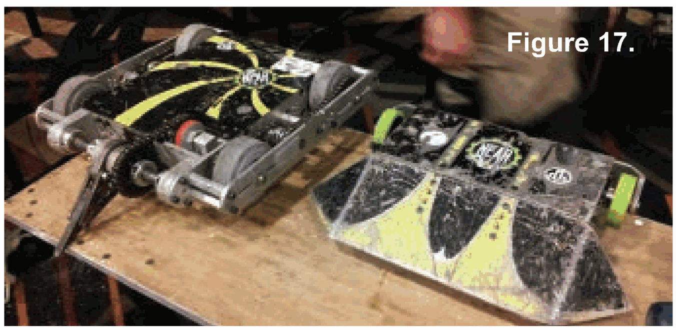


Figure 17.

DragonCon Robot Micro Battles 2013

Here's how the smaller bots fared at Robot Battles this year.

Thirteen Antweights and 15 Beetleweights competed in the Micro Battles event. Both classes had a diverse selection of machines competing for the win.

Among the Antweights were Quandry (**Figure 1**), a powerful horizontal spinner with a vibratory drive system; Ham (**Figure 2**), a foam armored robot; DDT (**Figure 3**), a destructive front mounted horizontal spinner; Reptar (**Figure 4**), the completely rebuilt defending champion with a front hinged flipping plate; Pad Thai Doodle Ninja (**Figure 5**), a classic four-bar flipper with a 3D printed chassis; Klazo (**Figure 6**), with a large vertical drum and extremely durable armor; and Kitty Catty Death Machine (**Figure 7**), a massive vertical spinner.

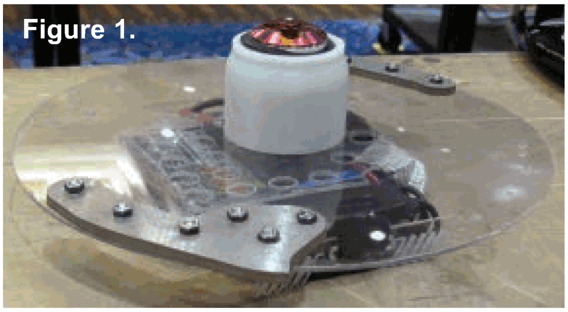


Figure 1.

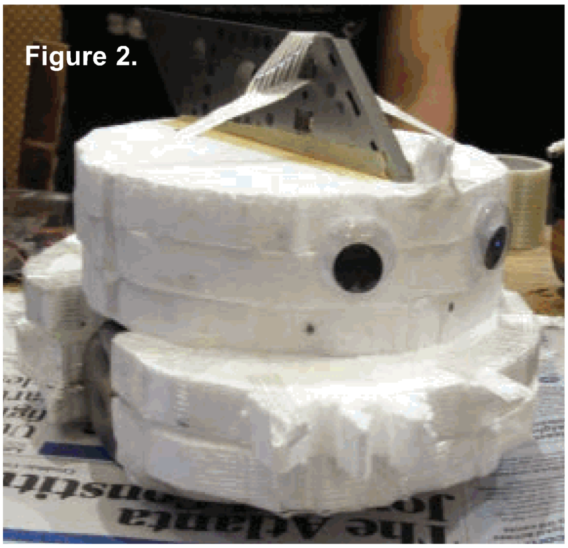


Figure 2.

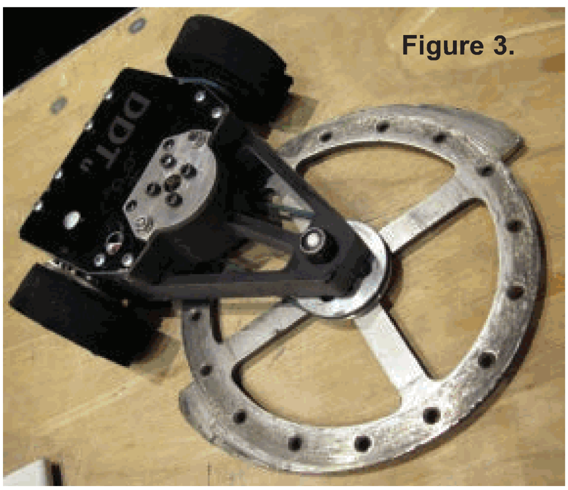


Figure 3.

Among the Beetleweights were a pair of robots armored with spinning Colson caster wheels aptly named Colson Powell (**Figure 8**) and ColsonBot Wedgee (**Figure 9**), a durable wedge and the defending champion; Flippy (**Figure 10**), a Weta kit (**www.kitbots.com**); Unskinny Bot (**Figure 11**), a front mounted bar spinner; The Hammer (**Figure 12**), an extremely powerful horizontal spinner with a welded tool steel chassis; Son of Magic Smoke (**Figure 13**), a massive thwackbot; Torgo (**Figure 14**), a lifter; Twisty (**Figure 15**), another horizontal bar spinner; Point of Fashion (**Figure 16**); a heavily armored wedge; and Sloppy P (**Figure 17**), a powerful 4WD vertical spinner.

Both classes fought in a single elimination tournament. With the pit and arena pushout, this meant a small mistake could take you out of the event. Some of the top competitors — including The Hammer, Sloppy P, and Reptar — all met their end by sending themselves through the pushout.

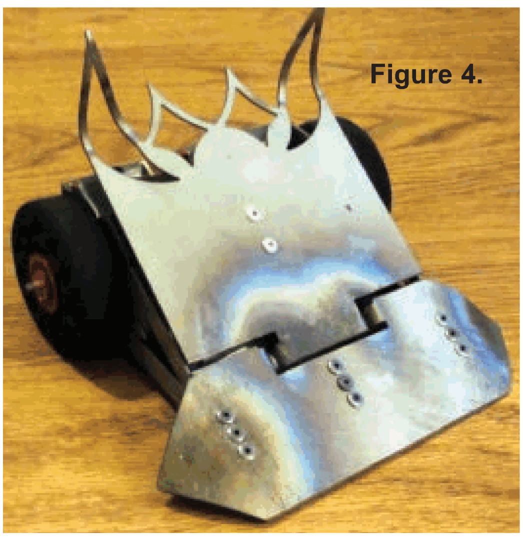


Figure 4.

The Antweight final four were Algos (**Figure 18**), Carbon Footprint, Segs, and Spiritual Void (**Figure 19)**. In the semi-finals, Algos eliminated Carbon Footprint and Spiritual Void eliminated Segs. Algos and Spiritual Void then fought three times before determining a winner, as the first two matches ran out of time.

The third match ended with three seconds remaining when Algos pushed Spiritual Void into the spinning pit

which (in the third round of a set of fights) counts as a pushout.

The Beetleweight final four were Family Joules (**Figure 20**), Flippy, Stability Margin (**Figure 21**), and Sloppy P. In the semi-finals, Family Joules eliminated Flippy and Stability Margin eliminated Sloppy P. Family Joules and Stability Margin then fought twice before a winner was decided. The first match ended in a double immobilization, and the second ended with Family Joules winning after Stability Margin ripped the weapon

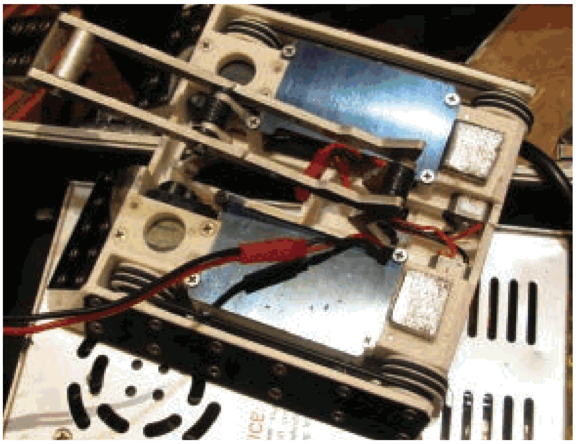


Figure 5.

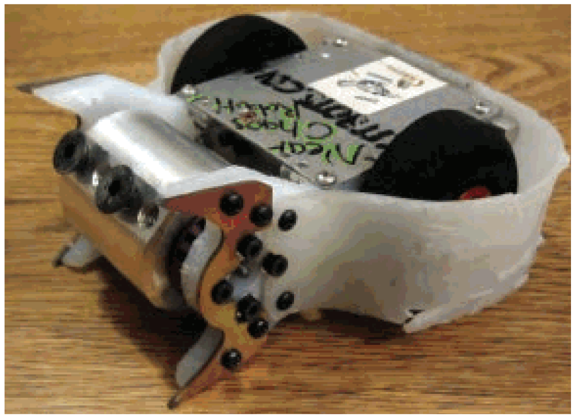


Figure 6.

Figure 7.


Figure 8.

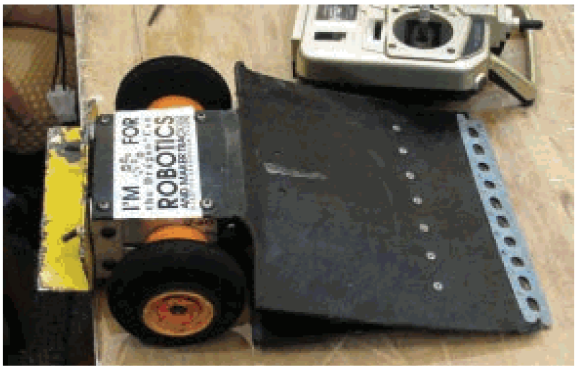


Figure 9.

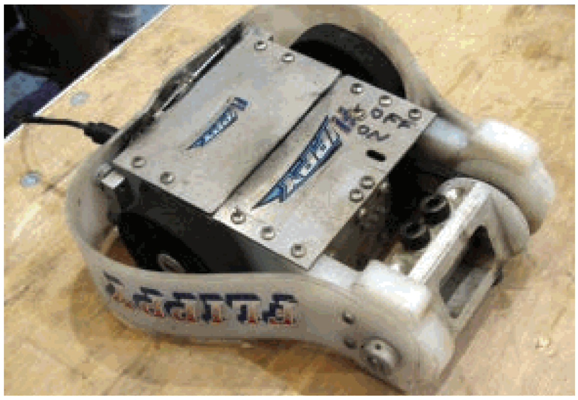


Figure 10.

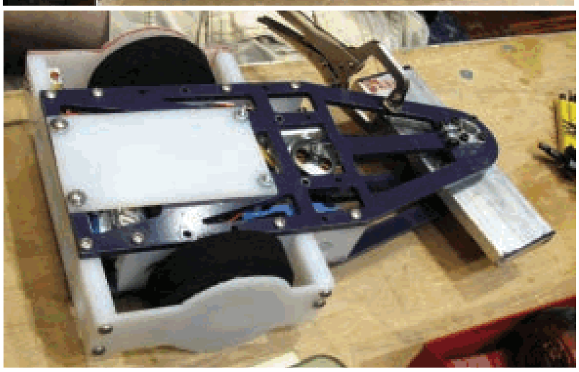


Figure 11.

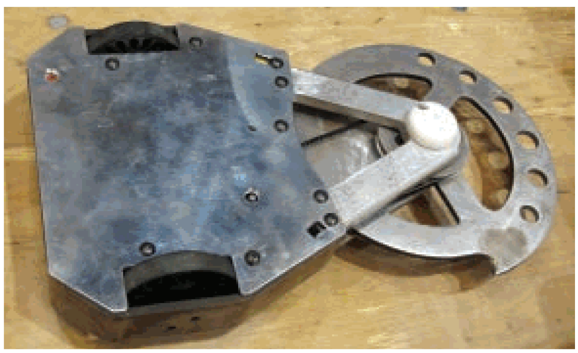


Figure 12.

and side armor of Family Joules off, which was shortly followed by Family Joules pushing Stability Margin into the pushout. **SV**

	Antweight	Beetleweight
1st Place	Algos	Family Joules
2nd Place	Spritual Void	Stability Margin
3rd Place (tie)	Carbon Footprint and Segs	Flippy and Sloppy P

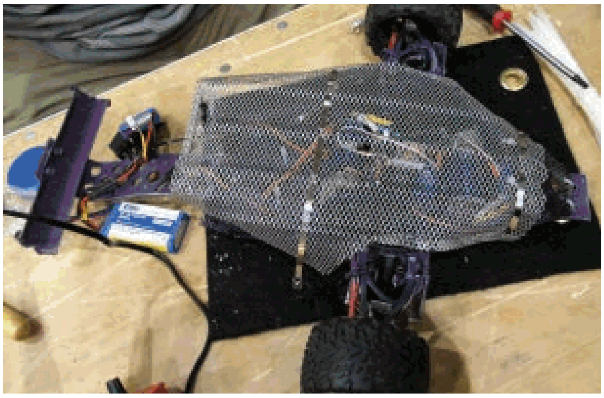


Figure 13.

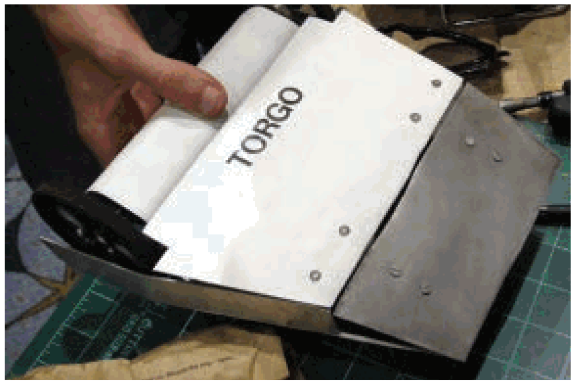


Figure 14.

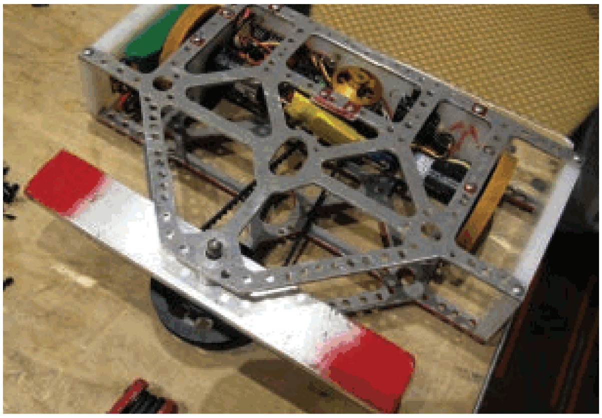


Figure 15.

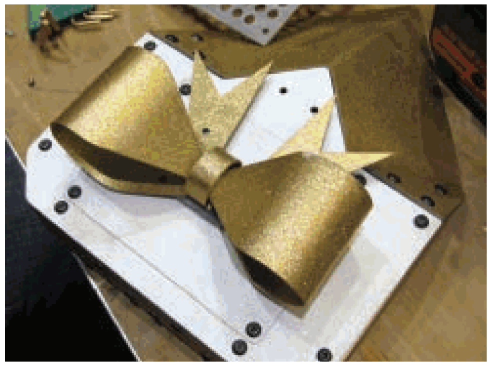


Figure 16.

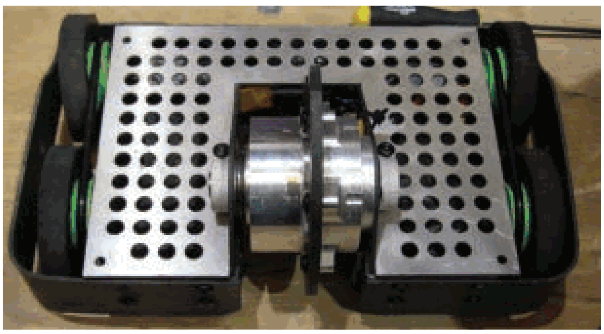


Figure 17.

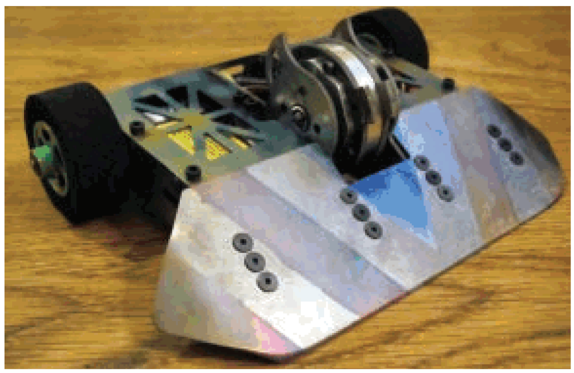


Figure 18.

Figure 19.

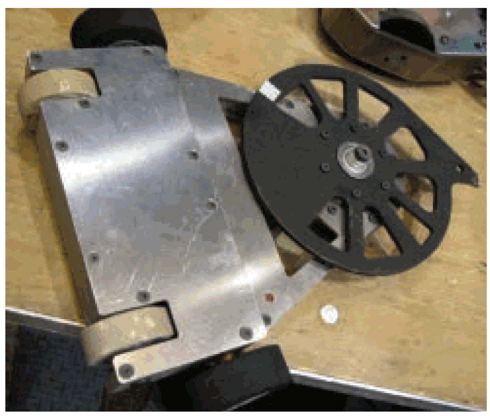


Figure 20.

Then and Now — A Decade Later With Carlo Bertocchini

● by Kevin M. Berry

This is the next installment in our series of articles about famous figures from a decade ago. 2002 ended the popular Comedy Central series, BattleBots™. 2003 inaugurated a new era in combat robotics, where our sport left the spotlight and we tried to fly on our own. Grassroots events sprung up everywhere, as documented in our 2012 series of articles "The History of Robot Combat." For 2013, we're taking a more personal approach, interviewing media stars from that time.

When it comes to the glory days of televised robot combat, few competitors can beat Carlo Bertocchini's record. He and his bots appeared in two Robot Wars and every BattleBots competition; and his heavyweight BioHazard is widely recognized as one of the classic bots of its time. Chosen by McDonalds for their immensely popular kid's meal series, BioHazard also appeared in the Tiger Electronics Pro Series of BattleBots toys. Carlo was widely respected in the builder community for his "always built, always ready" reputation when arriving at events.

Carlo graciously agreed to answer our questions about the "good old days" of 11 years ago.

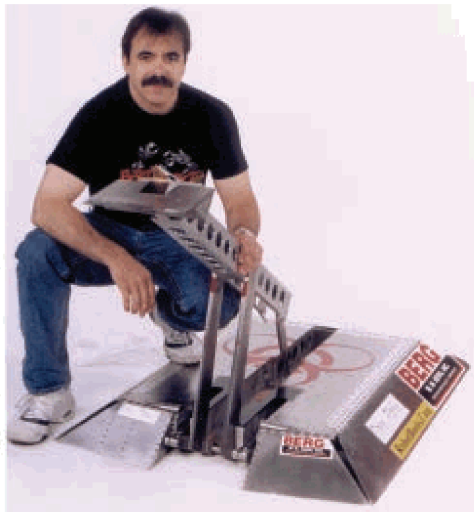
Combat Zone: Carlo, back in "the day," you did the big ones — Robot Wars and BattleBots. How did you get into the sport, and what hooked you so thoroughly you sunk so much time, energy, and money into participating?

Carlo: Marc Thorpe (founder of Robot Wars) showed up at the monthly meeting of the San Francisco Robotics Society of America (SFRSA). I believe this was in 1993. I had been a member for a while and I was involved with robot Sumo competitions at the time. He (Thorpe) showed us a cool robot arm he had been working on at Industrial Light & Magic. He also talked about his plan to stage a competition/art installation that would feature fighting robots. It sounded fun, so I started thinking about fighting robot designs.

It does take a lot of resources, but so do most things worth doing. I have done a little auto racing and I can tell you robot competition is a lot less expensive.

What makes it to TV is only a tiny fraction of what happens at an event, and what happens at an event is only a tiny fraction of what is required in order to prepare for the event. I probably spent an hour or two of work for each minute my robot or I was on TV. Others — who may have had just a single match televised — may have worked hundreds of hours for the privilege. That may sound like a questionable return for the effort, but not if you actually enjoy the process.

Consider the artist who spends his time creating a painting. The value of his activity is not measured by how many people see his art; it is measured in the joy of the creative process itself. The artist who does not enjoy painting and the robot maker who does not enjoy making robots should find other creative outlets.



CZ: You were at BattleBots during the big TV events. What was it like being a participant, from the first event to the last?

Carlo: A full answer would fill a book. For people who weren't there, a viewing of the television show only gives part of the story. Others who participated will remember the smell of burned rubber and electronics. They'll remember the din of the pits and the sound of the crashes in the arena. You worked feverishly on your robot while cameramen made your life difficult and B-list celebrities looked over your shoulder. We all had hopes of winning, but many of us also felt the peculiar camaraderie that happens when a group of like-minded people get together.

Readers of this magazine may be quite familiar with robotic sports, but back then it was new and it was seen as a little bizarre. There was a

sense that each of us was part of a spectacle that was poorly understood by most of the observers. In a way, we weren't 500 teams with 500 agendas; rather we were one big team trying to birth a big new baby.

CZ: You participated in the ComBots cup. Have you hit any of the other, non-televised events?

Carlo: I had been participating in the annual robot Sumo competition at the Exploratorium science museum before Marc's presentation at the SFRSA. In 1995, I spearheaded our company's involvement with the national FIRST robotics competition. I was captain of a team that included engineers and craftsmen, and about 15 students from the local high school. We built an awesome robot and we all went to Disney World for the competition. The best thing about that trip was the amazing amount of fun we had in Florida. The second best thing was that we actually won the competition and brought home the national title.

CZ: One of the things that got my kids into the sport were the toys. Let's face it, it's got to be cool to see your machine on McDonald's promotions. Did you go buy a Happy Meal just for the heck of it, hoping to get "your" toy?

Carlo: I didn't leave it to chance. I walked into the local McDonald's, explained the situation, and walked out with about 20 little BioHazards. These are now collector's items, worth thousands of dollars (I wish).

CZ: You started BattleKits, leveraging the most excellent BioHazard design. Others we've profiled in this article series (for example, Jim Smentowski) also used their high profile as part of a career change. Is robotics still your main thing?

Carlo: One of the obstacles I found in creating a world-class robot was the lack of suitable motors. After competing a couple years with less-than-ideal motors, I decided to do something about it. I purchased the best off-the-shelf motors I could find and I made several changes, including rewinding the armatures by hand. The performance of the robot generated so much interest from the other competitors that I decided to market the motors. That was the genesis of the AmpFlow company (**www.AmpFlow.com**).

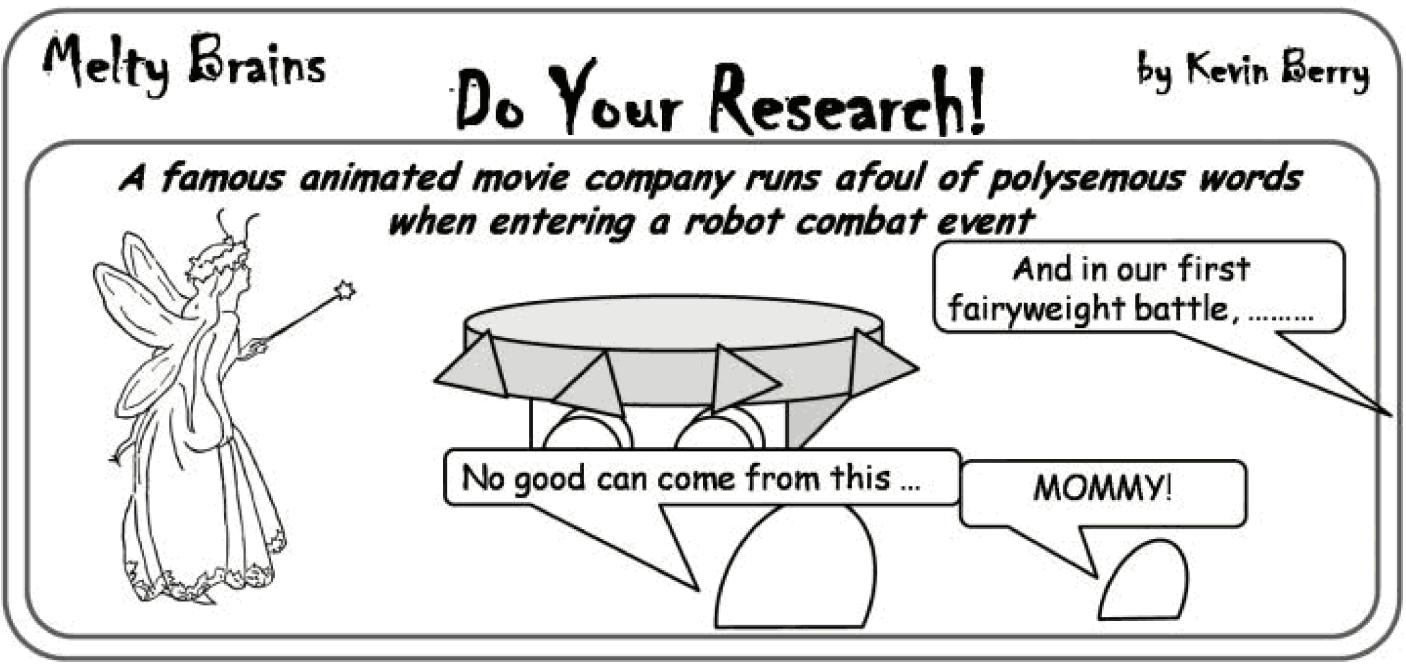
Now, many years later, in addition to the high-performance A28-150 and A28-400 motors, we offer several other lines of motors that have found their way into various consumer and industrial applications.

Creating the robot kits offered at BattleKits (**www.BattleKits.com**) was motivated in part by the sometimes daunting effort required to field a competitive robot. I designed a series of three robot kits that took some of the best features of my winning robot and made them available at a reasonable cost.

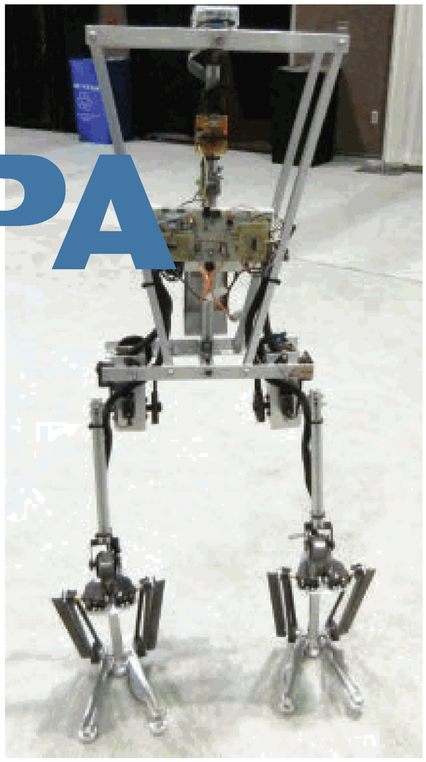
A team starting with a BattleKit equipped with AmpFlow motors can get into the game with a lot less time and effort. Over the years, we have shipped a lot more robots for non-sports applications — which was unexpected.

CZ: Any new ventures you'd like our readers to hear about?

Carlo: I have been more of a spectator than a competitor in the last few years, but I still have my finger on the pulse of the sport. I am often contacted by companies wanting to stage events, and even the occasional production company with plans for new robot-related TV shows. I have not found a good fit yet, but the readers of *SERVO* are welcomed to contact me with new ideas. **SV**



The Road to the DARPA Robotics Challenge



by Daniel Albert, Chris Mayer and Girts Linde

Go to **www.servomagazine.com/index.php?/magazine/article/november2013_Albert** to comment on this article.

Part 3:The Team

The famous American anthropologist Margaret Mead once said:

"Never doubt that a small group of thoughtful, committed people can change the world. Indeed, it is the only thing that ever has. ~ "

I started this biped project in 2006. I felt like I was climbing the highest mountain in the world on my own. This year, Chris and Girts joined up and we became a team. Now, I can see the top. This month's article is a team effort. Each member is working on different parts of the biped, so it is divided into the three sections: mechanical changes, the biped systems, and the Gazebo simulation.

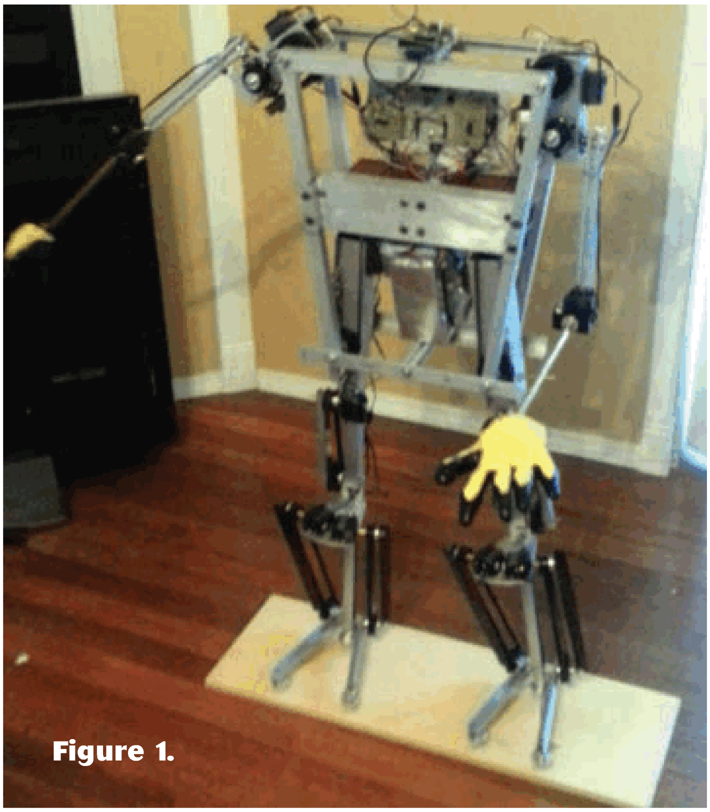


Figure 1.

Mechanical Changes — Daniel

Finding servos that are high torque that don't require you to take a second mortgage on your house is not an easy task. Our original servos claimed over 400 oz/in of torque, but they just could not deliver those numbers. We have just placed an order with a company in China that has a giant size R/C servo for under $100 with a claimed 850 oz/in of torque.

In the meantime, we have made some significant changes in the biped. Using the Gazebo simulator, we finally came up with some hard torque numbers. Strong servos are crucial for two joints in particular. The hip in/out and the knee servos need about 1,600 and 1,200, respectively, to lift and hold the upper body weight.

The first change we made was to replace one of our original servos that could not hold the hip in place with a super strong albeit heavy 3,200 oz/in Torxis servo from Invenscience. Unlike hobby servos, I was not able to back-drive these. This works out nice when the biped is standing still or when that joint doesn't need to move.

These are powerful servos built with strong output shafts. I can simply shut them down and they hold their positions. At under $300 each, I would have used them for

the whole biped if they didn't weigh so much.

The supporting brackets for these servos were a bit tricky to fabricate. It was necessary to keep the weight high in the upper torso for an inverted pendulum effect. The belts for the pulleys also needed to be aligned with the existing hip pulley. You can see the two 1.5 inch aluminum tubes used to hold the servos in **Figure 1**. With only 1/16 inch wide walls, these tubes are very light and very strong. Supporting these tubes required replacing the four torso vertical u-channels with one inch square tubes. Again, these tubes are very light and very strong.

Once the upper torso was strengthened, it was time for the arms. We wanted to simulate the movement generated by swinging arms. The old legs proved perfect for this. They were too weak as legs, but as arms they work well enough even with the original servos. When the new servos arrive, it should be easy enough to swap them out.

Only the knee servo mount needed to be modified to allow the forearm to be attached. We couldn't let the arms have no hands, so as a joke we put a couple of gloves on Watson. So, did this new hardware improve the design? You bet. We charged up the batteries and ran a simple program to stand and move the arms. This was mostly a test of our battery capacity. Was there enough for the huge power drain? Did our redesign of the legs create a more stable platform? How long would the battery pack last?

We were pleasantly surprised that the robot ran for about six minutes before collapsing. (Watch the video at **www.dropbox.com/sh/dbgt3j7w9u8oold/iphsLQwkkT**.

The next step will be the actual first step. We feel confident that the new servos will have the additional torque to get us walking. The DARPA challenge test that we want to concentrate on will be Task 4: "Open Door, Enter Building." This will require a real hand. Full articulation is not necessary since all three doors will have lever handles.

Biped Systems — Chris

Servos

These are probably the most important component. Like many other bipeds, Watson uses 16 servos for primary movement. There are a few additional servos used for the pan and tilt of the cameras and head rotation. The arms and legs are where we need the most power. It is important that the servos match the torque requirements for each joint.

Higher torque allows for better movement, but they tend to weigh more. We are continually changing servos to get the best balance of torque and weight for each joint. The servos also consume the majority of the power. The servos often go from idle to full power very quickly. This causes power spikes. Isolating the servo power supply from the electronics prevents voltage drops and possible unwanted resets.

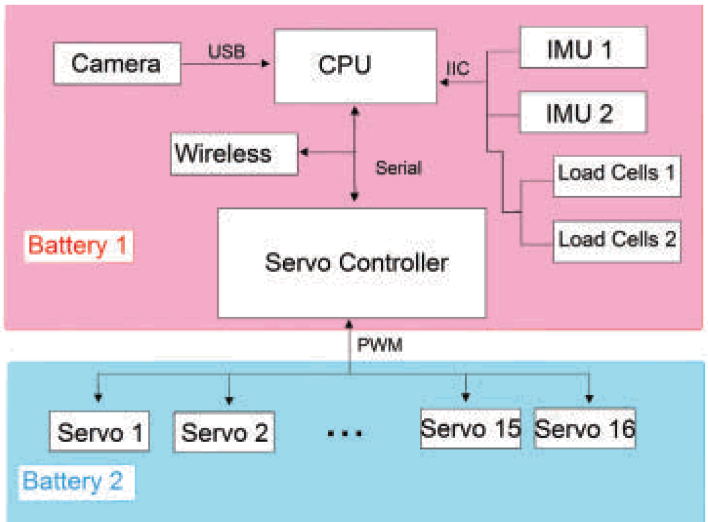
Servo Controller

Servo controllers come in a wide variety of choices. The one we prefer is the standard Lynxmotion SSC-32 controller. The SSC-32 has timing commands in addition to the simple position commands. For example, the command "#5 P1600 T1000<cr>" will move servo 5 to position 1600 and take one second to get there, regardless of how far the servo has to travel. We have added a power and ground rail to the back of the board to allow greater current for our high power servos. We found it is sometimes an advantage to use more than one servo controller. This way, we can split up the power required and use a smaller group of servos. The SSC-32 uses a serial interface, making it easy to interface to our CPU.

Electronics

The biped's electronics are fairly straightforward and use a collection of mostly off-the-shelf parts.

We have two custom boards to use for our load cells and the auto-balance feature of each foot. We also have a hand-built power board that manages important things like a safety power-off switch and fuses.



CPU

The CPU is the biped's brains. We have used everything from Arduinos and Raspberry Pis, to ODROIDs. Each has advantages and drawbacks, but for the most part it really doesn't matter as long as you can control all of your electronics and do all of your processing in real time.

The Arduino was fine to start out, with its I^2C bus and multiple serial ports. When we needed to connect a camera, we had to move over to the Raspberry Pi. This also has both an I^2C bus and serial ports, and in addition has USB ports and a special camera connector.

When we needed more processing power to process stereo cameras for mapping and navigation, we moved up to the ODROID. All of our code is in C, so it recompiles easily and runs on all of the above platforms.

IMUs

The IMUs — or Inertial Measurement Units — give

motion feedback to the biped. The standard IMUs provide three vectors for linear acceleration, rotation, and the surrounding magnetic field. In the past, these were individual sensors but most new chips are integrated together.

The benefits of the integrated package are reduced size and cost, as well as requiring less wiring. Sensor fusion is the key to a good IMU, so we have our own "secret sauce."

Load Cells

In addition to the IMUs, Watson uses three load cells on each foot. These provide pressure sensitive feedback to make it much easier to know where the center of mass is located in relation to the feet. Almost all other bipeds rely only on the IMUs to keep balance. Watson is able to maintain a stable stance much easier. Rather than wait for the IMU to alert the biped that it has already started to fall, Watson can keep its center of mass over the feet *before* it starts to tip over.

Wireless

The wireless serial interface serves multiple purposes. One purpose is to send commands to the biped's CPU. These commands can be anything from "Walk Forward One Step" to "Make Me A Sandwich." The amount of autonomous behavior versus full control is dependent on the software.

Another and just as important purpose is telemetry and feedback. Again, this is dependent on exactly what you decide to get back. Everything from low level IMU and load cell data, to a video feed from the camera will help in debugging and controlling the biped. XBee is our choice for both the Arduino and Raspberry Pi. It allows control via a simple serial port and requires very few resources. We have USB ports and Wi-Fi adapters for optional control over the Internet.

Camera

A stereo camera gives us a full 3D reconstruction of everything in its field of view. With a full color image, we recognize objects, do facial recognition, and map out our biped's surroundings. We use proprietary hardware and software that allows us to use inexpensive cameras.

Batteries

One of the most difficult parts of building a non-tethered biped is the power source. Only in the last five years has the battery industry been able to supply us with enough energy density to build an all electrical biped that can last long enough to perform even the simplest of tasks.

Higher voltages are typically better for generating more torque from a servo motor. Higher C or Capacity values are needed to provide the sudden need for power when the torque requirements get high.

We prefer the newer LiFePo4 battery chemistry that produces voltages around 3.2 rather than the 3.7 of the LiPo. This lets us run around 12 volts for most servos.

Gazebo Simulation - Girts

One of the problems we have is figuring out what servos to use. We use the Gazebo simulator to estimate the required torques in the joints. Gazebo is a 3D multi-robot simulator for outdoor environments. It can simulate physical interactions between the robots and the world, as well as the sensor feedback for the robots. In fact, it was used in the first phase of the Robotics Challenge where the teams programmed a simulated Atlas robot to complete the trial tasks.

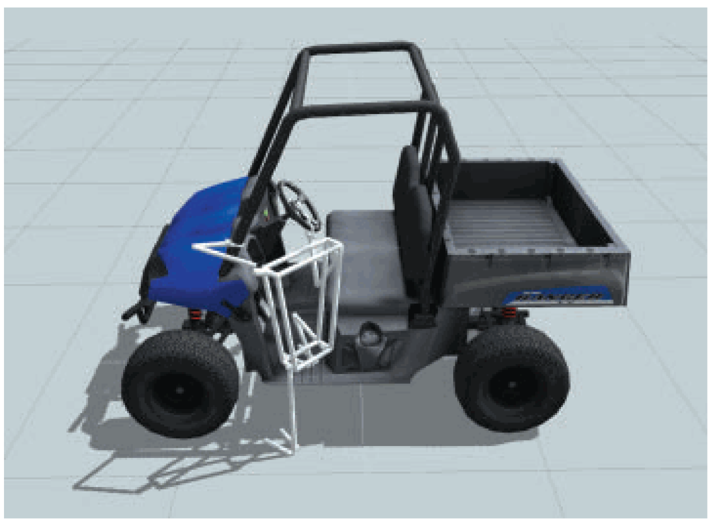
Gazebo is an open-source project — one of the projects of the Open Source Robotics Foundation — so it is available to everyone. The environment and objects used for the virtual trials are available as well, so we can download virtual goodies like a fire hose or a buggy to use in simulations with our robot.

To run a simulation, Gazebo needs a file describing the structure of the robot and some C++ code to control the robot. At this point, we have a robot model consisting of 27 rigid parts and 26 joints between them. In addition to that, we have some 200 lines of code to play animations using PID controllers to produce the individual torques for each of the joints. This mimics the use of digital servos that have PID controllers inside to control their position.

As a byproduct of that, we can see the amount of torque required at each of the joints at each point through the animation sequence. As expected, just standing straight requires the least amount of torque. Though, while moving, the torques can be as large as 25 N*m in the knee and 15 N*m in the hip for the current weight of our robot.

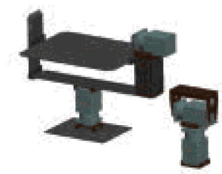
So far, we use just a part of the simulator's abilities. In addition to movement, it can simulate input from various sensors like IMUs, cameras, and load cells. If that works well, we'll be able to work on the algorithms without the risk of breaking the real robot.

Stay tuned for more updates on our biped project. **SV**



The Actuator Solution for Full Scaled Robots

DYNAMIXEL PRO



Data		Model	H42-20-S300-R	H54-100-S500-R	H54-200-S500-R
		Unit	Data	Data	Data
Rated voltage		V	24	24	24
No load speed		RPM	28.3	35.2	35
No load current		A	0.61	1.06	1.18
Continuous operation	Speed	RPM	15.59	32.7	32.1
	Torque	Nm	5.596	21.142	39.131
	Current	A	1.989	5.930	9.505
Resolution		Step/turn	304,000	502,000	502,000
Gear ratio		-	304	502	502
Backlash		arcmin	3.5	3.5	3.8
Interface		-	RS-485 / CAN	RS-485 / CAN	RS-485 / CAN
Operating temperature		˚C	5~55	5~55	5~55

All in One-Actuator

High Power, High Precision

Up to 200W BLDC motor and up to 500,000 pulse per turn (4096 absolute resolution)

Full Modular Solution

Ease of implementation (no additional design for parts needed)

Ease of maintenance (simple part swapping)

Sophisticated Control Algorithms

Position and speed input commands with dual-loop current control

Novel Gear Reduction System

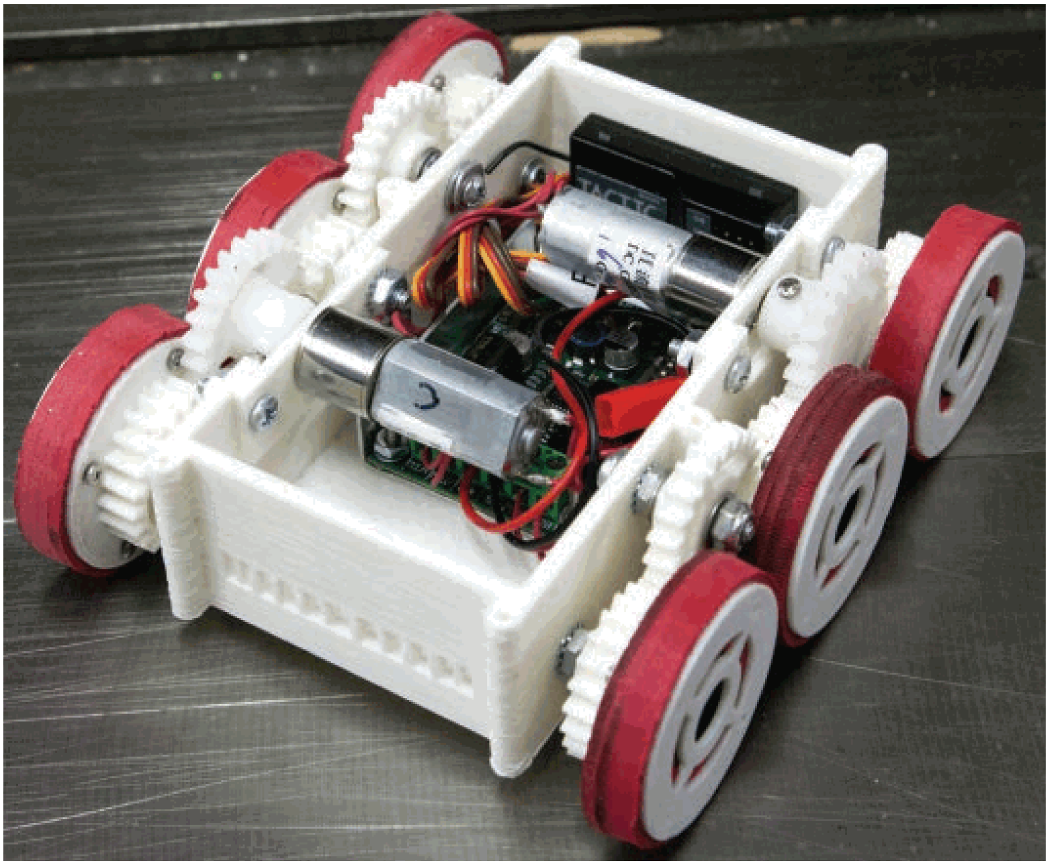
High torque output, light weight, with high impact resistance

USA E-mail : america@robotis.com / Tel : +1-949-333-3635
JAPAN E-mail : japan@robotis.com / Tel : +81-3-4330-3660
Etc. E-mail : contactus2@robotis.com / Tel : +82-70-8671-2609
DYNAMIXEL are registered trademarks of ROBOTIS Co., Ltd.

www.robotis.com

Build the Plastic Bot of Destruction

Part 1



By Michael Simpson

Post comments on this article and find any associated files and/or downloads at **www.servo magazine.com/index.php?/magazine/article/november2013_Simpson**.

In this series, I am going to take you through the design and creation of a plastic battling robot — complete with weapons. This series is not about engineering or doing things a specific way. It's all about fun. I have created a working prototype, so I know what components will be used for the bot. In the next stage of design and construction, I want to invite suggestions from the readers.

History First

Before I get into the details, I feel it's necessary to understand the history that led me down this path.

A couple years back, I had four booths at the DC Science and Engineering Fair. I was asked to build a demo version of my KRMx02 CNC. Additionally, I wanted to create a couple battling bots (built with the CNC) that the spectators could control. I had seven months to prepare, so I had plenty of time to experiment.

Frank

My first attempt was a robot called Frank, short for Frankenstein. This robot (**Figure 1**) had a lot of smarts. It had a full blown PC on board running Windows 7. It could be monitored via Wi-Fi, and was controlled via a 12 channel radio. The problem with Frank was that he was long on brains, but short on brawn. (You can see him in action at **www.youtube.com/watch?v=mBnxZLw7tVA** and **www.youtube.com/watch?v=ggEZR2L3kxA**.)

Frank was easily defeated by another prototype I had built. While he did not have any weapons, his size, shape,

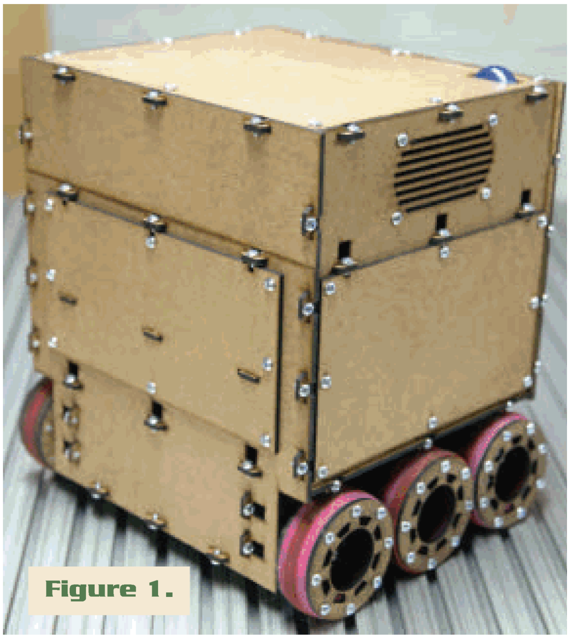


Figure 1.

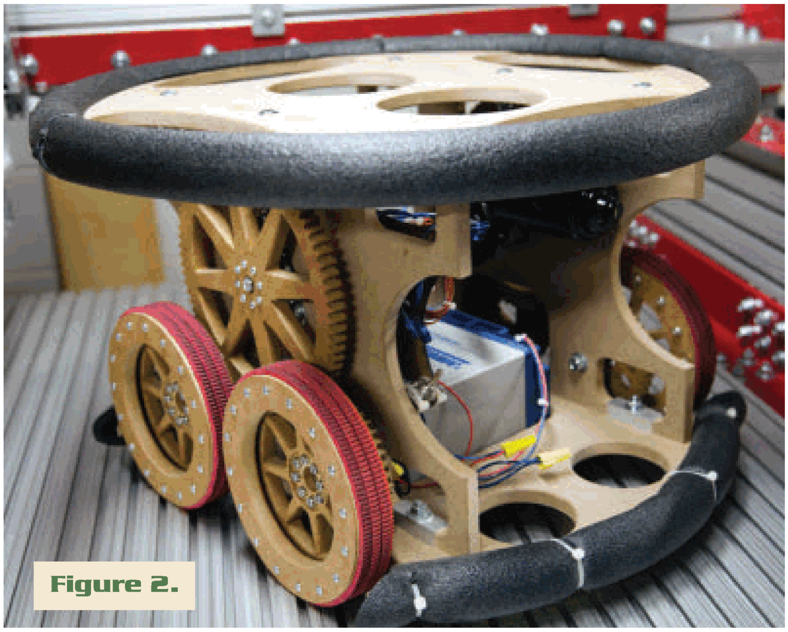


Figure 2.

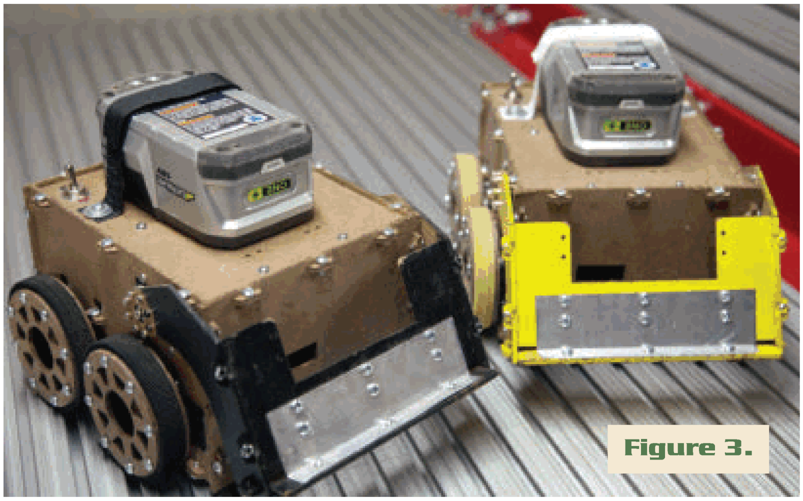


Figure 3.

and complexity was not a good fit for a thousand screaming kids. Frank didn't make the cut.

Rover

My second attempt was a larger bot called Rover (**Figure 2**). He was originally going to be an automated bot, but I thought he could be turned into a fighting robot with a few changes to his initial design. (See him in action at **www.youtube.com/watch?v=VOH7AckxQWc** and **www.youtube.com/watch?v=GzgmcDcgyuk**.)

The problem with Rover was that he was too large. I would have to devote a very large area to the arena, and put up protective barriers so the kids couldn't get hurt. This proved to be too much. Rover would not be making the trip.

PROCS

I went back to an early design and made some basic changes. What I came up with was a bot called PROCS (**Figure 3**). PROCS stands for "Precision Robotic Cube Sumo." (You can see them in action at **www.youtube.com/watch?v=Is1Nn1LK_AQ** and **www.youtube.com/watch?v=k1VnUkrzC2U**.)

I built a 5' x 5' arena that could be easily transported and had small sides to keep the bots from flipping out of it. The arena itself could be placed on a couple of tables which would be at a good height for spectators. PROCS would be going to the fair.

The Show

The show turned out to be a great success (**Figure 4**). Hundreds of kids had a blast. We had a line of kids waiting to play with the bots from the opening until closing each day of the event. Our booth crew had to take shifts to manage the contests.

The bots held up very well considering. They lost a lot

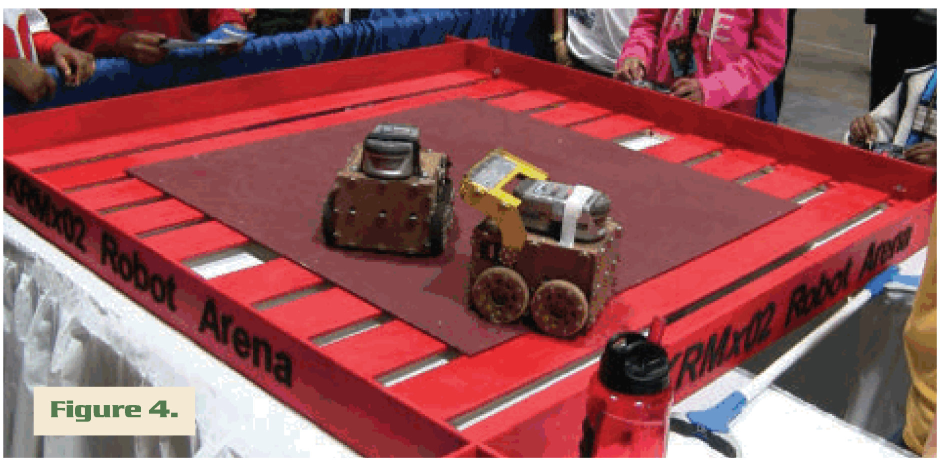


Figure 4.

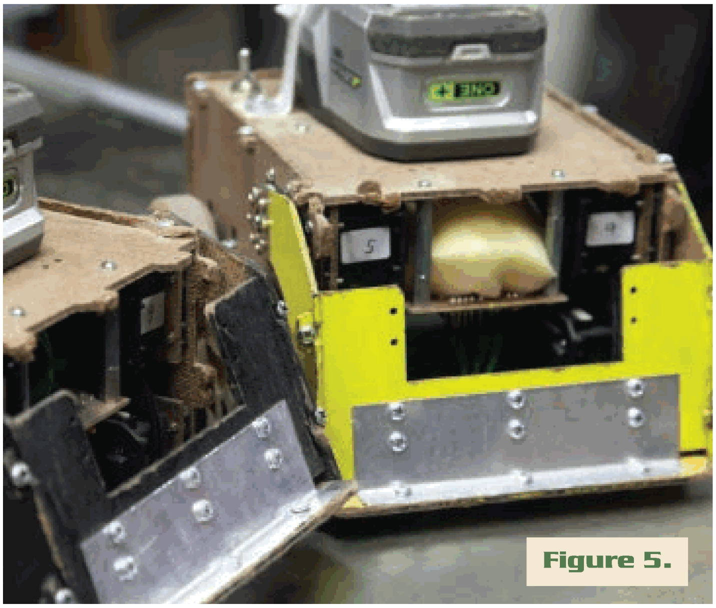


Figure 5.

of screws that held the front panels on, and eventually the front panels were lost (**Figure 5**). However, the bots battled on. I observed that every time a screw popped off, the crowd would cheer. When a front panel was lost, they went wild. It was pandemonium. I would tape the front panels in place just so they could be torn off again. Nothing is cooler than a bot dragging around part of its ripped off chassis.

One thing I took away from this experience is the idea of designing a set of bots that could rip each other apart. The PROCS design looks like a good place to start, but in this application it's really not. Because of the actuators and microcontroller control, each cost over $2,000 to build. I also thought they held up a little too well. I went back to the drawing board.

What I came up with is what I call PBOD. This stands for Plastic Bot of Destruction. The opening photo shows a prototype that I have been working on, but I'm sure we can make improvements. Let's take a look at how this series will unfold.

Early Testing

The prototype shown at the beginning was a proof of concept. Could I design and build a small inexpensive bot that could be used in a radio controlled battle scenario? The prototype showed it could be done. All of the parts except for the electronics, hardware, and tires were completely 3D printed. Although it's a little sturdier than what I eventually want, it helped me come up with a set of goals for this project.

Goals

1. The PBOD has to be printed in ABS or PLA. If machined, it has to be expanded PVC.
2. The frame cannot be made as a unibody frame. It must be constructed with a base, with separate sides and top.
3. The parts can be 3D printed, laser cut, or machined with a CNC mill.
4. It must use a motor controller that is capable of mixing the RC signals to control a differential drive with two channels.
5. A four-channel radio should be able to control the robot and weapons.
6. No projectile weaponry. No fire or explosives. Flippers, spinners, or hammers are okay.
7. The wheels must fall into the same rules as the frame. With the exception of the tread material, they can be foam rubber, rubber, or some other soft material to allow for traction.

These goals are only a starting point; once the weaponry is complete and the PBOD has a few battles under its belt, we can make changes.

You can see the prototype in action at **www.youtube.com/watch?v=pK57Tw3dL3U.**

Gear

Now that I have a working prototype and a set of goals, I have chosen the gear I will use to construct the PBOD.

Controller

I decided on the Sabertooth dual 12 amp motor driver from Dimension Engineering (**Figure 6**). While 12 amps is overkill for the bot we will be building in this series, it allows for a great deal of expansion in the future.

Features:
- 6V to 25V input (2s to 6s LiPo)
- Synchronous regenerative drive
- Ultra-sonic PWM switching
- Thermal and over-current protection
- Lithium protection mode
- Built-in RC mixer

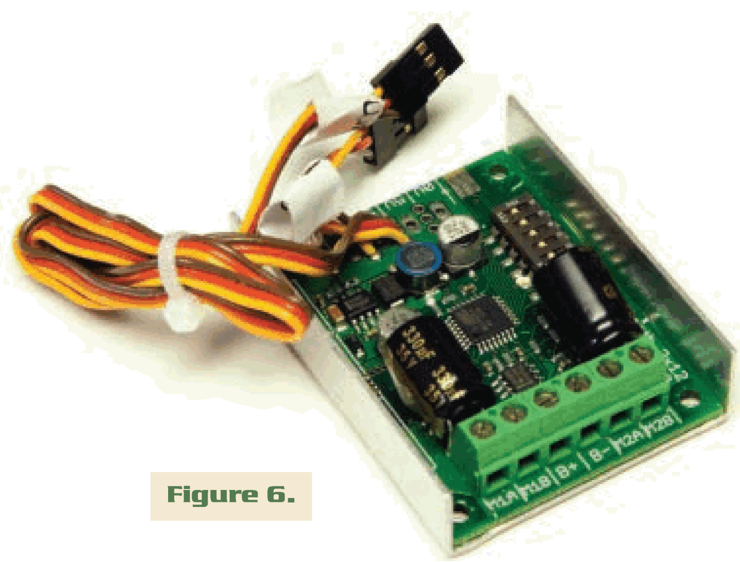
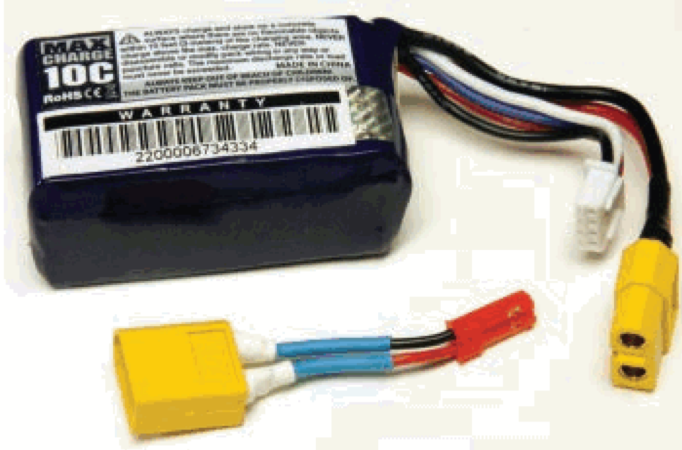


Figure 6.

This controller is designed for a 1-12 pound combat bot. It can run at 12 amps per motor with a peak of 25 amps. The hookup could not be simpler: Two connectors plug into your RC receiver; your motors and battery connector attach to the provided terminal strip.

Figure 7.

Figure 8.



Motors

The motors shown in **Figure 7** are 300 RPM 12V motors. They have a 600 mA rating with no load. They are a perfect match for our PBOD when using a 3s LiPo. The motors don't come with wires attached, so we will add them.

Figure 9.

Battery

For the motors mentioned previously, a small 850 mA three-cell LiPo (**Figure 8**) is a perfect match. It gives us a long run time and is tiny enough to fit into the smallest of space. It also has enough to power our weapon system. A small adapter may be needed, depending on the controller hookup we choose.

Radio

For the RC radio, I'm using the Tactic radio system. This is an inexpensive 2.4 GHz RC radio system. The standard receiver for both the four- and six-channel transmitters is the six-channel TR624 shown in **Figure 9**. For the transmitter, I'm using the radio shown in **Figure 10**.

You may have noticed it's a Futaba radio. Tactic has a device called an AnyLink. It's designed to allow you to use an older radio (any brand) with a Tactic receiver. It connects to your transmitter via the trainer port. In my case, it plugs directly into my old six-channel 72 MHz radio (**Figure 11**). I have removed the original antenna and crystal, so it's only using the controls on the radio.

The AnyLink is a six-channel transmitter, so when it is plugged into a six-channel radio, you gain access to all the channels.

Figure 10.

Figure 11.

In truth, you can use just about any 2.4 GHz system. If, however, you have an old analog radio, the Tactic receivers are only $20 each and the AnyLink adapter is only $25.

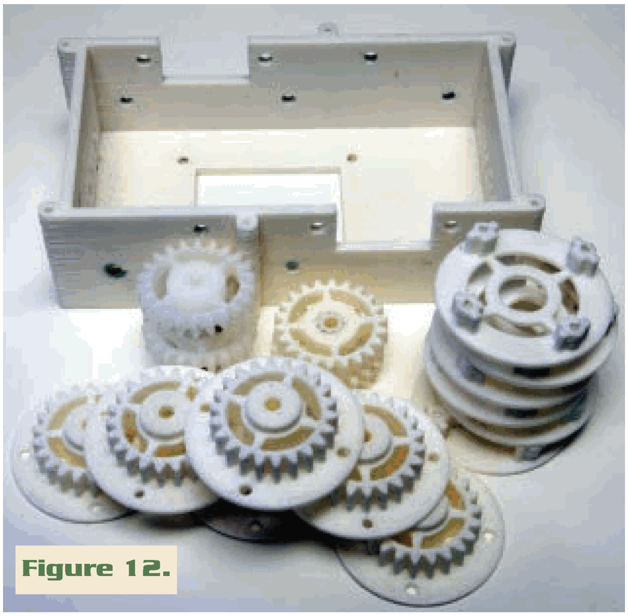


Figure 12.

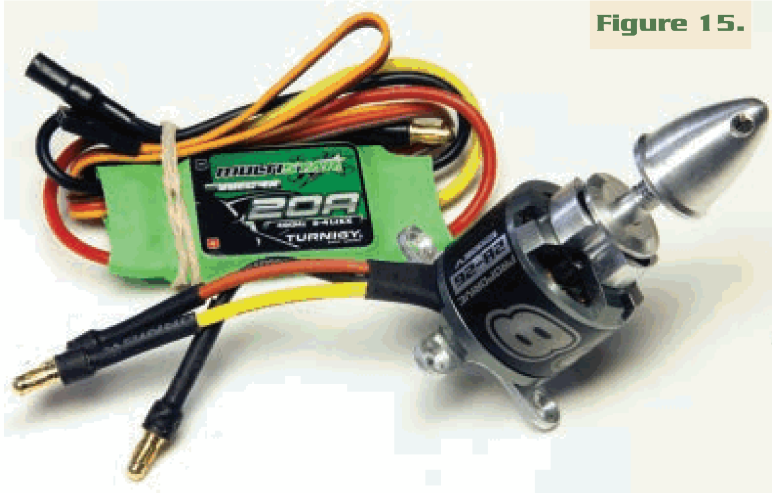


Figure 15.

Frame and Wheels

The parts used on the prototype are shown in **Figure 12**. In the final design, the front and sides will be separate components. The other components make up the drive gears and wheels. Originally, the tires were made from laser cut craft foam, shown in **Figure 13**. I think I can come up with something a little easier to make.

The hardware that hold the gears, tires, and controller in place is shown in **Figure 14**. This is mostly #6 and #4 hardware. The hardware will change as the design is updated. Much of it is used to hold the wheels and idler gears.

Figure 13.

Figure 14.

Weapon Gear

For the weapon, I plan on using the 20 amp brushless speed controller and motor shown in **Figure 15**. These should work well for a spinner weapon. For a flipper, a servo may work better. Perhaps a combination of both.

One idea I want to try out is a high speed weed whacker type spinner. I'm working on a brushless motor controlled weed whacker that will attach to one of my hex copters. This will be used to cut fruit down from a very tall pear tree. I plan on a version of this weapon for the PBOD.

Conclusion

I will be starting a webpage that will list the complete bill of materials for all the items in this series. Once the design is complete, I will post the STL files for you folks that want to print your frame and wheel parts. I will be posting the prototype parts, as well.

You can find the PBOD web page at **www.kronos robotics.com/pbod**.

Please be sure to post questions and suggestions in the *SERVO Magazine* forums at **http://forum.servomagazine .com/viewtopic.php?f=49&t=17029**.

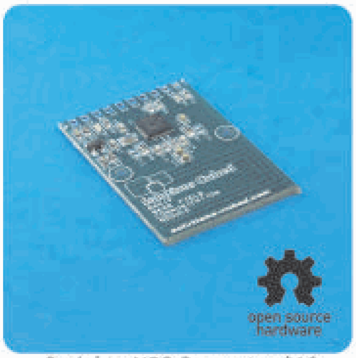
Next month, we will start the new design and take a look at some wheel alternatives. At the very least, I want to have the basic frame parts available for assembly.

I will also look at machining some of the parts on my CNC machines so that those of you without a 3D printer can participate in this series. **SV**

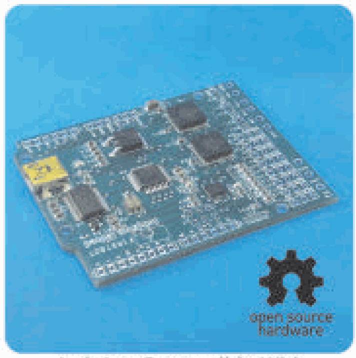
Try Our

Open Source Hardware Modules

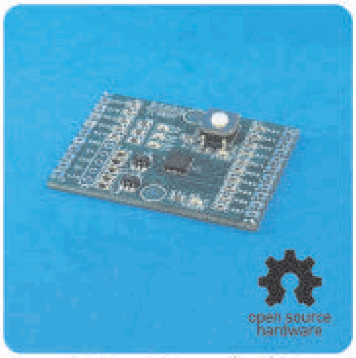
In Your Next Design



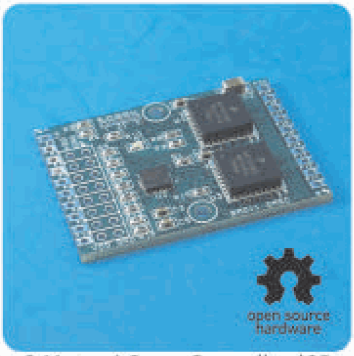
Serial to NFC Converter $16



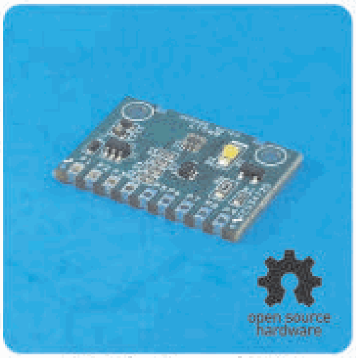
Arduino Compatible With Motor Drivers $35



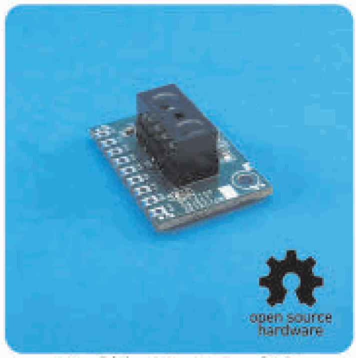
10 Servo Controller $18



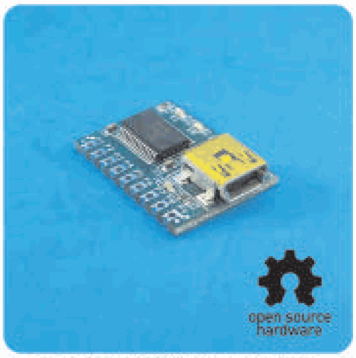
2-Motor 4-Servo Controller $25



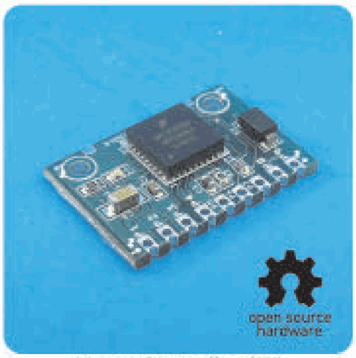
I2C Color Sensor $14.50



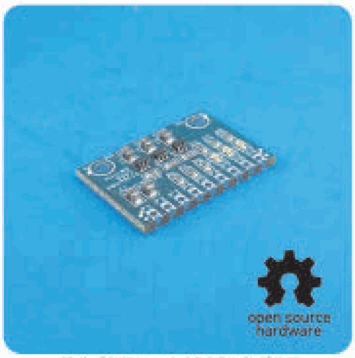
IR Object Detector $10



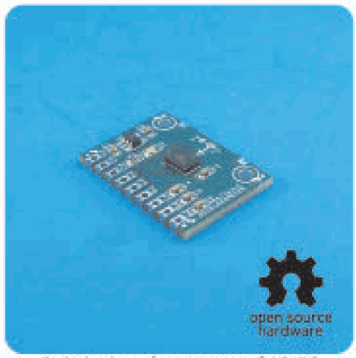
Serial to USB Converter $15



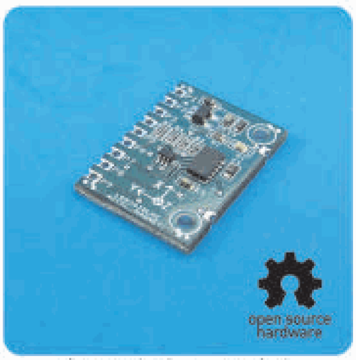
Motor Controller $15



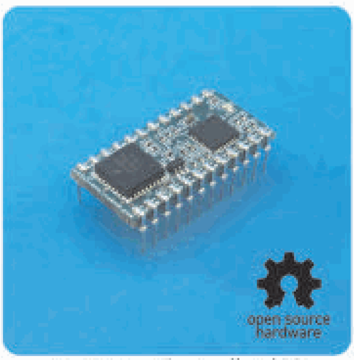
Triple Power Switch $13



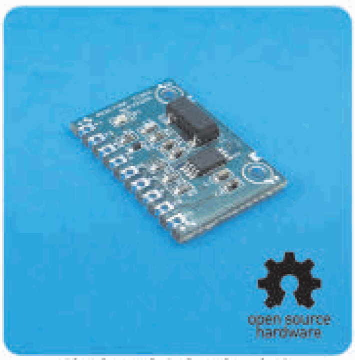
3-Axis Accelerometer $12.50



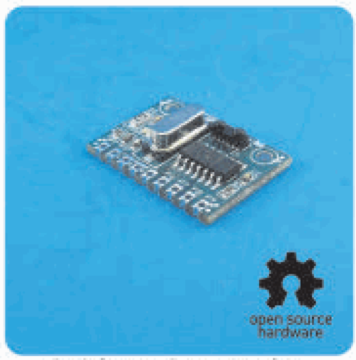
Electronic Compass $18



PID Motor Controller $30



Clock and Calendar $12

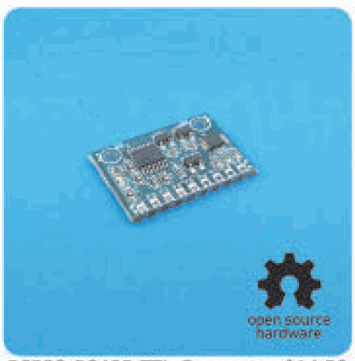


Serial to IR Converter $15

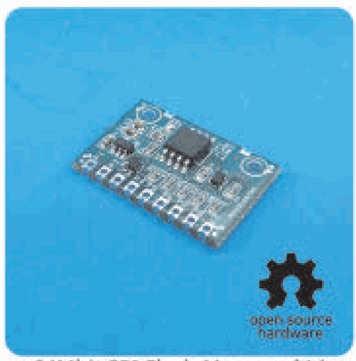
hardware made easy

Solutions Cubed

designservices@solutions-cubed.com
www.solutions-cubed.com



RS232-RS485-TTL Converter $14.50



64Mbit SPI Flash Memory $14

Get Your Motors *Runnin'*

The very first experience I had with a hobby servo involved a 555 timer. I had a friend that was deep into RC aviation. One day, we were sitting in his shop working on a plane and I noticed that he kept picking up the transmitter to test his servo linkage hookups. Not only did he have to fire up the transmitter, he had to make sure the newly installed servo was attached to the receiver. It looked painful to me. So, I suggested that I create a small box that could drive the servo directly without the need of a transmitter and receiver. After some intense reading (no Internet in those days), I figured out how to make a 555 timer and a single-axis potentiometer-based joystick do the job of an RC transmitter and receiver.

By Fred Eady

Post comments on this article and find any associated files and/or downloads at **www.servomagazine.com/index.php?/magazine/article/november2013_Eady**.

Digilent, Inc.
Digilent Motor Shield
Uno32
MPIDE
Gear Motor
www.digilentinc.com

When an RC servo is forced to move, it makes a very distinctive sound. As long as there are humans walking the earth, there will be a need for wheels, motors, and RC servos. With that, this month's discussion will revolve around driving RC servos and DC motors using a Digilent motor shield mounted on a Uno32.

Uno32 Revival

The Uno32 smiling in **Photo 1** had its original factory bootloader overwritten. In that the Digilent motor shield was originally designed to run under the control of firmware generated by MPIDE, it might be a good idea to restore the Uno32's factory bootloader image. The re-image process is pain free.

First, I downloaded the Uno32 bootloader image from the Digilent website. The second step involved importing the downloaded bootloader hex image into MPLAB. Once the bootloader hex file was imported, I used a PICkit 3 to Flash the replacement factory bootloader image. To verify that the bootloader Flash completed successfully, I used MPIDE

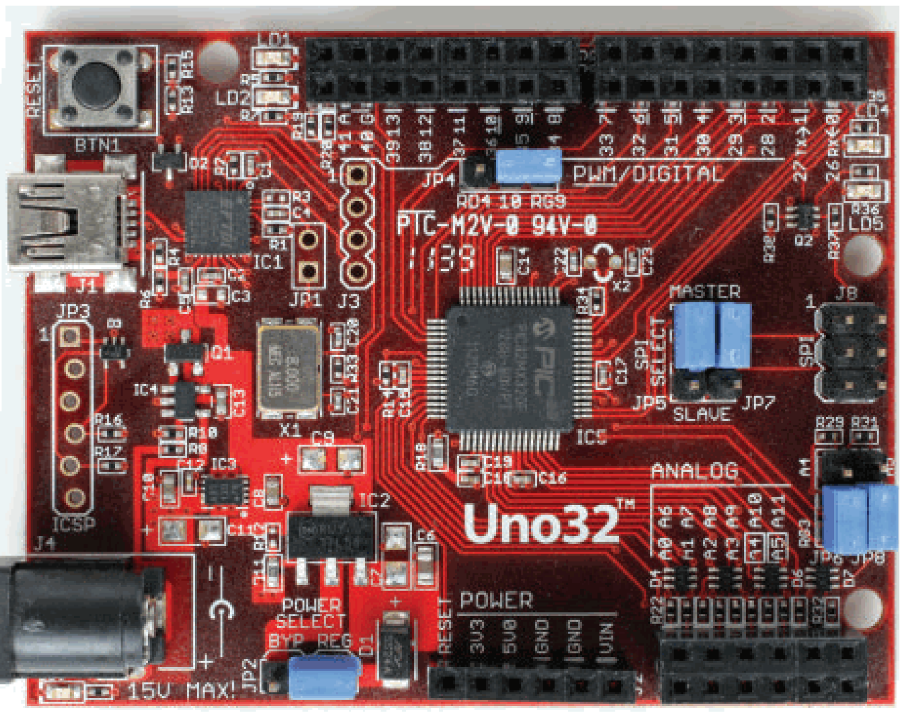


Photo 1. The name pretty much says it all. The Uno32 is based on a Microchip PIC32MX microcontroller and is capable of running many of the original Arduino sketches.

to load and execute a simple LED blinky program. The Uno32 is back in business.

The Digilent Motor Shield

The key to understanding how to use the motor shield pictured in **Photo 2** is to have a firm grasp on the hardware components that make it up. Please reference **Photo 2** and the motor shield schematics as we walk through its electrical subsystems.

DC Motor Drive Subsystem

As you can see in **Schematic 1**, the motor shield is based on the Texas Instruments DRV8833 dual H-bridge motor driver. The DRV8833 is capable of driving one stepper motor or two DC motors. You can see this a bit more clearly in **Figure 1**.

Basically, the output pins AOUTx and BOUTx logically follow the AINx and BINx input pins. The logical relationship that exists between the DRV8833's input and output pins is illustrated in **Figure 2**. The reverse and forward modes are what one would expect.

Let's talk about the coast and brake modes. Coasting occurs when the H-bridge is shut down. In coast mode, the recirculation current is allowed to flow through the MOSFET body diodes. The MOSFET body diodes are the reverse biased diodes located across each MOSFET's drain and source. Brake (slow decay) mode does not allow the recirculation current to flow through the MOSFET body diodes by shorting the motor winding.

The motor speed can be

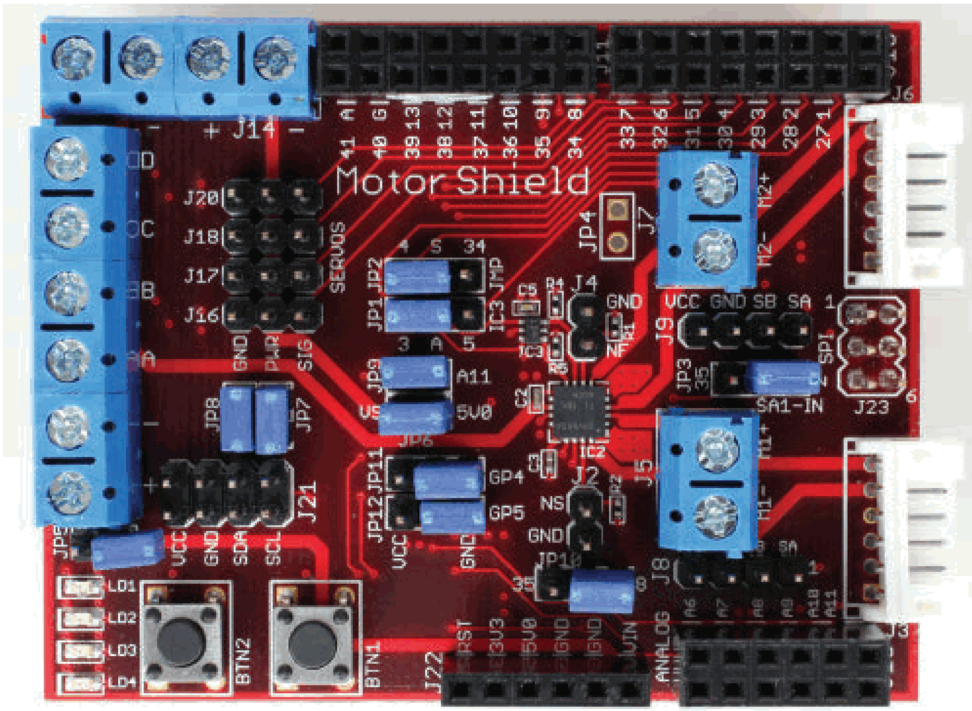
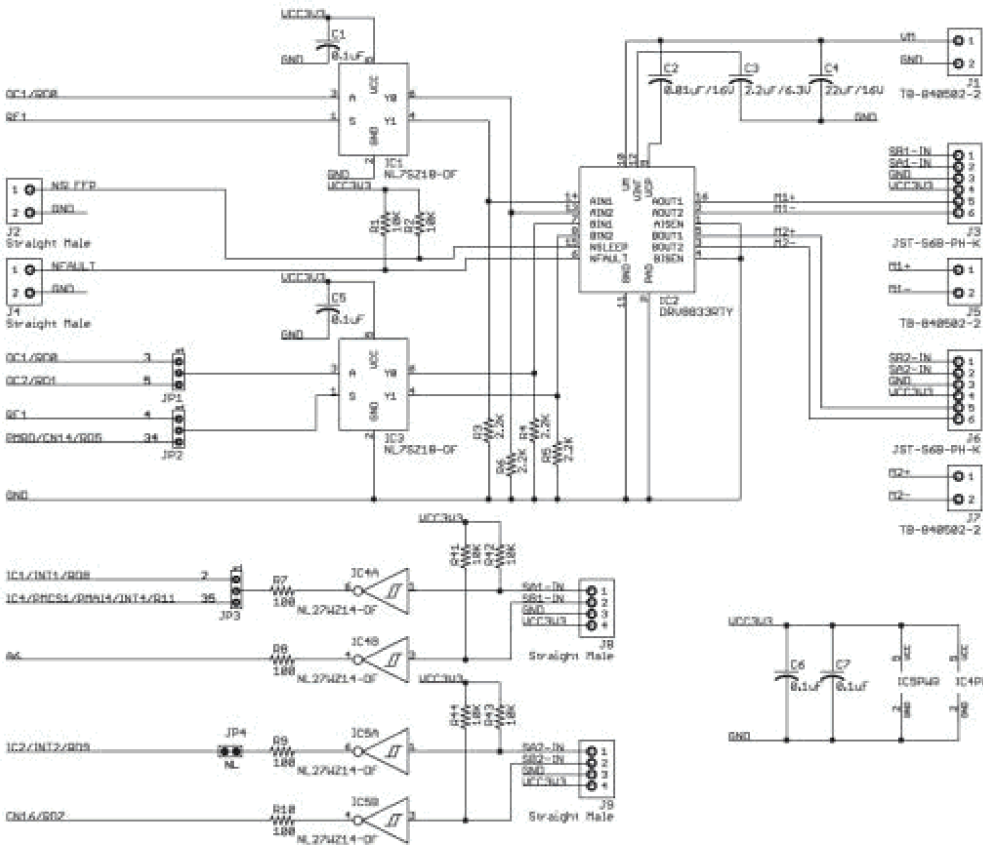


Photo 2. The Digilent motor shield can drive up to two DC motors or one stepper motor. There are also pins to accommodate up to four RC servos. An I2C expander gives the user an additional two pushbuttons and four LEDs.



Schematic 1. This was obviously designed to accommodate all of the features of the Digilent gear motors. However, it is a flexible design that is also capable of driving just about any small two-wire DC motor.

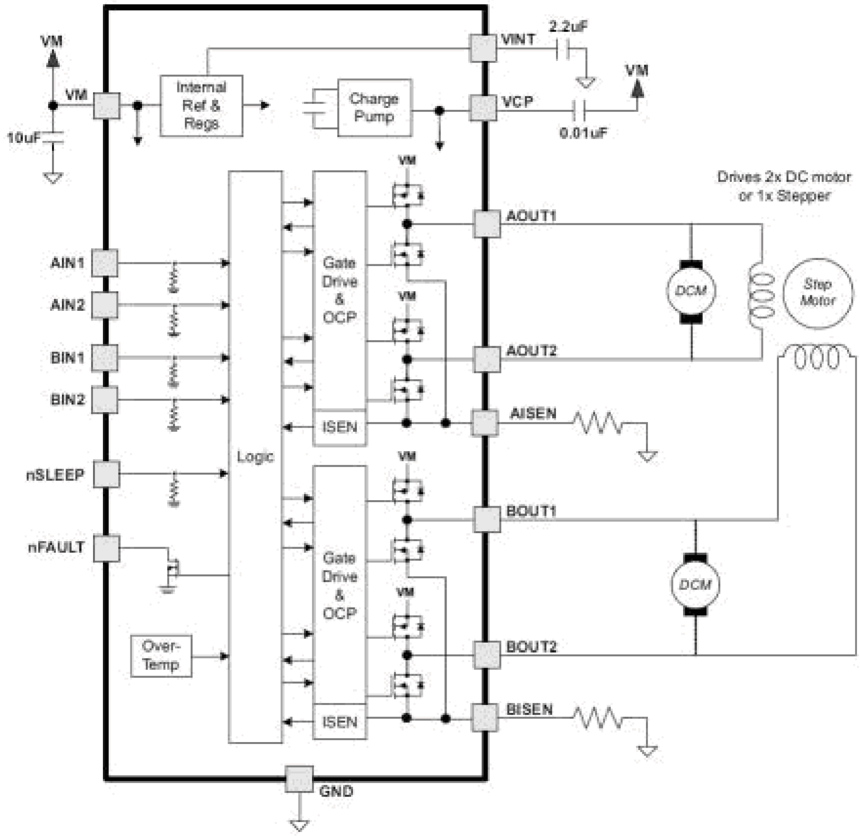


Figure 1. **There are plenty of words describing the DRV8833 in the datasheet. Once again, a picture is worth 1,000 of them.**

controlled by applying a pulse width modulation (PWM) signal to the DRV8833 input pins. Naturally, the PWM mode also allows the attached motor or motors to be reversed. The PWM logic table for the DRV8833 is contained within **Figure 3**.

xIN1	xIN2	xOUT1	xOUT2	FUNCTION
0	0	Z	Z	Coast/fast decay
0	1	L	H	Reverse
1	0	H	L	Forward
1	1	L	L	Brake/slow decay

Figure 2. The relationship between the input pins and output pins is rather obvious. However, it may not be intuitively obvious that fast decay disables the H-bridge and allows the recirculation current to flow through the MOSFET body diodes. Slow decay mode shorts the motor winding.

xIN1	xIN2	FUNCTION
PWM	0	Forward PWM, fast decay
1	PWM	Forward PWM, slow decay
0	PWM	Reverse PWM, fast decay
PWM	1	Reverse PWM, slow decay

Figure 3. Now that you know what the decay modes consist of, this too becomes a logical table of operations.

Input		Output	
S	A	Y_0	Y_1
L	L	L	Z
L	H	H	Z
H	L	Z	L
H	H	Z	H

Figure 4. The automatic high impedance state change allows the resistors to force their associated demultiplexer input pin logically low.

Let's shift our attention to **Schematic 1**. The DRV8833 AINx and BINx inputs are all tied logically low via resistors R3-R6. The DRV8833 AINx and BINx pins are being fed from a pair of 1:2 demultiplexers. The logic level at the A input of each multiplexer is passed to the Y0 or Y1 demultiplexer output, depending on the logical state of the select pin. The Yx output pin that is not selected reverts to a high impedance state. In that the DRV8833 inputs are pulled logically low, it is imperative that the deselected demultiplexer output not influence the DRV8833 input pin. The NL7SZ18 1:2 demultiplexer truth table is shown in **Figure 4**.

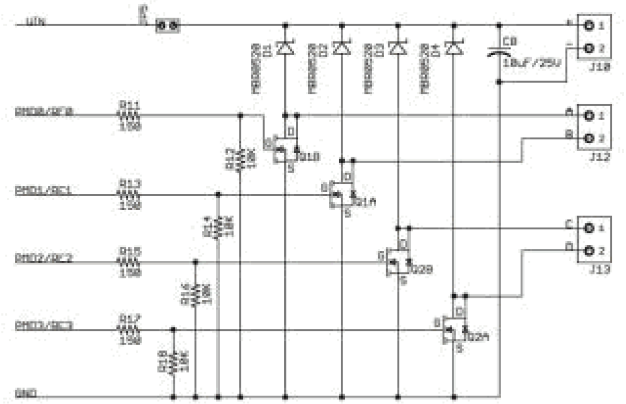
We can close out our DRV8833 input circuitry examination by observing that shorting J2 will put the DRV8833 to sleep and disable the H-bridge circuitry. We can also say with certainty that if we manage to overcurrent, overheat, or undervoltage the DRV8833 outputs, an active low fault signal will be presented at pin 1 of J4. If the fault continues to occur while driving a single motor with a single output, we have the option to parallel the DRV8833 outputs which will increase its output current handling capability. The DRV8833 can drive motors in the voltage range of 2.7 to 10.8 VDC. The motor drive voltage must be supplied at J1.

We can also put a cap on the walk through of the DRV8833 output circuitry. The Digilent gear motor resting in **Photo 3** generates those SAx quadrature encoder signals you see at J3 and J6. The quadrature-formatted encoder signals are used to indicate speed and direction of the motor shaft. The raw quadrature-encoded signals are buffered by IC4 and IC5. The quadrature encoder signals are optional, and a standard two-wire DC motor can be driven without the assistance of the SAx quadrature encoder signals using J5 and J7.

As you can see, the DRV8833's AISEN and BISEN current sense pins are both grounded. This indicates that the motor shield design is not taking advantage of the DRV8833's current monitoring pins.

Stepper Motor Driver Hardware

This subsystem does not depend on the DRV8833.



Schematic 2. **There's not much to say about this circuit. However, if you don't drive a stepper motor with your motor shield, you can use the MOSFETs as high current solid-state switches.**

Photo 3. **This Digilent gear motor interfaces directly to the motor shield via J3 or J6. A pair of Hall-effect sensors produces a quadrature signal as the motor shaft rotates.**

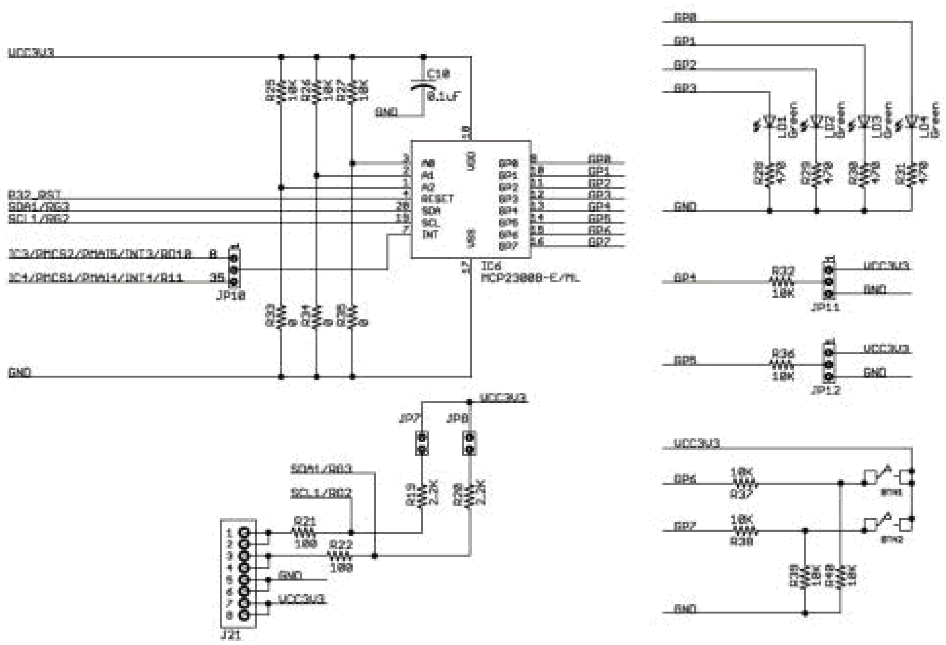
Instead, the stepper motor drive electronics consist of four open drain MOSFETs and four associated steering diodes. The stepper motor driver hardware is graphically depicted in **Schematic 2**. The quad of open drain MOSFETs can also be used as independent solid-state switches. The stepper motor driver subsystem can be powered externally by removing JP5 and supplying power via J10.

Servo Drive Subsystem

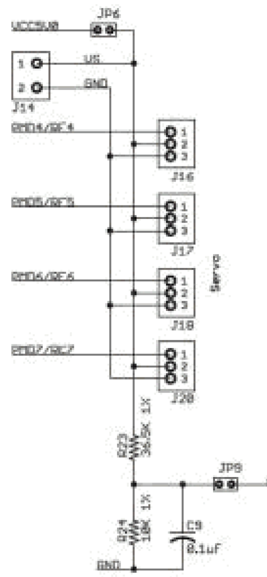
The servo drive subsystem consists of a couple resistors and a capacitor. Hobby servos are driven with a logic level signal. Thus, the servo drive motor is the only power consuming element. This is reflected in **Schematic 3**, which details four servo positions. Resistors R23 and R24 form a simple voltage divider that allows the servo power source voltage to be monitored by the host microcontroller. Like the stepper motor driver subsystem, the servo drive subsystem can be powered externally or from the host 5 VDC rail.

I²C Expander Subsystem

This is a very interesting part of the motor shield. The center of this subsystem is a Microchip MCP23008 I²C expander. The expander IC you see in **Schematic 4** is nothing more than an eight-bit I/O expander that is



Schematic 4. **The motor shield takes advantage of the Uno32's I2C bus and employs the services of the MCP23008 to add LEDs, pushbuttons, and a pair of user inputs.**



Schematic 3. **Again, there is not much to discuss. The magic that positions each servo is performed by the host microcontroller.**

configured and controlled using the I²C protocol. The MCP23008's eight I/O pins are configured as inputs on high nibble and output on the low nibble. The low nibble drives a quad of LEDs. Pushbuttons are attached to the two most significant bits of the high nibble. GP4 and GP5 are dedicated to jumper switches that provide a logical high or logical low input.

Digilent Motor Shield Firmware

The motor shield can be driven using the SoftPWMServo and Wire libraries. However, there is also a dedicated MotorShield library that takes the pain out of communicating with the MCP23008. Let's see what it takes to spin some motor shafts and blink some LEDs using the I²C bus.

Driving Miss DC Gear Motor

Referencing the chipKIT *Motor Shield Reference Manual*, we find that the J3 motor interface's Arduino enable pin (Enable1) is Uno32 pin 3. The associated direction pin (Direction1) is assigned to pin 4. We will use the SoftPWMServo library to place a PWM signal on the Enable1 pin:

```
#include <SoftPWMServo.h>

// Uno32 Enable and Direction Pin Assignments
const int pEnable1 = 3;
const int pDirection1 = 4;
```

The motor control logic behind the enable and direction versus PWM signals is contained within **Figure 5**. We'll need a place holder for the PWM value, and we'll also need a logical switch to toggle the Direction1 pin:

```
int pwmVal;            // PWM value
bool bToggle;          // forward/reverse switch
```

Okay. Now that all of our variables and pins are defined, let's go ahead and initialize them. While we're at it, let's initiate a serial link so we can use the serial monitor as a debugger:

```
void setup()
{
    pinMode(pEnable1, OUTPUT);
    pinMode(pDirection1, OUTPUT);
    bToggle = true;
    Serial.begin(9600);
}
```

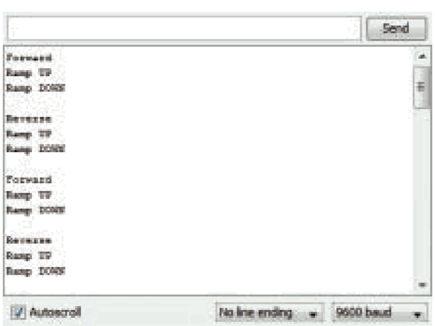
The plan is to accelerate the gear motor, decelerate it, change directions, and repeat the whole process again:

```
void loop()
{
    switch(bToggle)
    {
      case true:
        digitalWrite(pDirection1, LOW);
                                  // forward
        Serial.println("Forward");
        delay(1000);
      break;
      case false:
        digitalWrite(pDirection1, HIGH);
                                  // reverse
        Serial.println("Reverse");
        delay(1000);
      break;
    }
     Serial.println("Ramp UP");
    for(pwmVal= 0x00;pwmVal < 0x0100;pwmVal++)
    {
       SoftPWMServoPWMWrite(pEnable1, pwmVal);
                                  // send PWM value
      delay(50);
    }
      Serial.println("Ramp DOWN\r\n");
    for(pwmVal=0xFF;pwmVal > 0;pwmVal—)
    {
      SoftPWMServoPWMWrite(pEnable1, pwmVal);
                                  // send PWM value
      delay(50);
    }
    bToggle = !bToggle;
}
```

Our little motor twister program checks the condition of the *bToggle* variable and determines how to set the logical value of the Direction1 pin. The gear motor is then ramped up and ramped down in the selected direction. The *bToggle* variable gets inverted and we perform the ramp up/ramp down in the opposite direction. To check things along the way, the direction and ramp status is sent out to the serial monitor which is captured in **Screenshot 1**. Power consumption with the motor running is 70 mA.

Positioning the Servo

Again referencing the motor shield manual, we find



Screenshot 1. This display serves two purposes. I use it to notify me of the progress of the program and to verify that the motor was really doing what I told it to do.

DIR1	EN1	Result
0	0	Stop
0	1/PWM	Forward
1	0	Stop
1	1/PWM	Reverse
DIR2	**EN2**	**Result**
0	0	Stop
0	1/PWM	Forward
1	0	Stop
1	1/PWM	Reverse

Figure 5. This table is valid for steady state and PWM motor control.

that Servo1 and Servo2 are accessed via Uno32 Arduino pins 30 and 31, respectively. Oh, heck. We'll enumerate all four and we'll reuse the SoftPWMServo library:

```
#include <SoftPWMServo.h>

const int servo1 = 30;  // Servo 1
const int servo2 = 31;  // Servo 2
const int servo3 = 30;  // Servo 3
const int servo4 = 31;  // Servo 4
```

The library function call *SoftPWMServoServoWrite(pin, position)* will move the servo motor associated with the Arduino pin number to a position represented by microseconds. The extent of a typical hobby servo is 1,000 µS to 2,000 µS, with center at 1,500 µS. So, to center Servo1, we would substitute 30 for pin and 1500 (1.5 mS) for position. Typing out the servo library call could get to be a pain. To keep our fingers sane, let's wrap the servo movement library call into a couple of C functions:

```
static void centerServo(int servonum)
{
    SoftPWMServoServoWrite(servonum, 1500);
}

static void moveServo(int servonum, int pos)
{
     SoftPWMServoServoWrite(servonum, pos);
}
```

As you can see, the *centerServo* function simply commands the servo to center its shoe. If we need to arbitrarily move the servo shoe, we place a call to the *moveServo* function. Let's go ahead and perform the servo Arduino pins setup:

```
void setup()
{
  pinMode(servo1,OUTPUT);
  pinMode(servo2,OUTPUT);
  pinMode(servo3,OUTPUT);
  pinMode(servo4,OUTPUT);
  Serial.begin(9600);
}
```

Once again, we'll fire up the MPIDE serial monitor for the same reasons we did before. The code will be very easy to read since our servo function calls are self-documenting:

```
void loop()
{
  centerServo(servo1);
  Serial.println("Centered");
  delay(2000);
  moveServo(servo1,1000);
  Serial.println("1000");
  delay(2000);
  centerServo(servo1);
  Serial.println("Centered");
  delay(2000);

  moveServo(servo1,2000);
  Serial.println("2000");
  delay(2000);
}
```

The *delay* function calls provide way more time than required for the servo shoe to complete the predetermined moves that are programmed in. The serial monitor output for our servo application is shown in **Screenshot 2**.

Mastering the I²C Expander

The I²C expander IC can be manipulated with the Arduino wire library. When you get your homebrew wire code, you will come to the realization that you have spent a bunch of time reinventing the MotorShield library code.

The format of an I²C exchange between the Uno32 and the MCP23008 begins with a start bit, followed by a control byte. The control byte encapsulates the slave address and the R/W (Read/Write) bit. The most significant seven bits of the control byte contain the slave address.

Take another look at **Schematic 4**. The MCP23008's address lines A0, A1, and A2 are all pulled to ground with ZeroΩ resistors. Thus, one would say the I²C address of the MCP23008 is zero (000). Wrong!

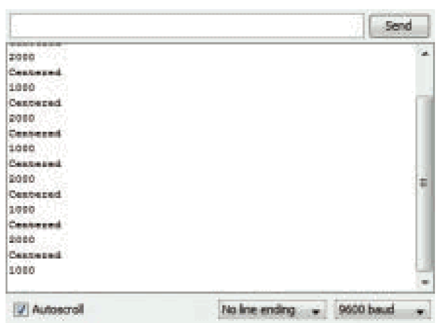
The MCP23008 datasheet tells us that the MCP23008 address bits are located in the three least significant bits of the seven-bit slave address field. The upper four bits of the MCP23008 slave address are 0100. This is a bit confusing as a discussion. So, let's map it out:

```
S 0100 A2 A1 A0 R/W
Where:
       S = Start bit
       Slave Address = 0100000
       R/W = Read/Write bit
The Slave Address is partitioned as 010 0000 or
0x20.
```

The 0x20 gets you to the MCP23008's front door. The next thing you better have is the room address. The "room" address is really one of 11 MCP23008 register addresses. Once you enter the right room, you can deliver that pizza in the bag you're carrying. Here's what that front door, room, and pizza look like in the *MotorShield.h* file:

```
// unique I2C device address
 #define DEVADDR     0x20

 //Register Addresses
 #define IODIR        0x00
       // Data direction register (setting pins
       // for input/output)
 #define IPOL         0x01
 #define GPINTEN      0x02
 #define DEFVAL       0x03
 #define INTCON       0x04
 #define IOCON        0x05
```



Screenshot 2. **I've moved my share of servo shoes. This has to be the easiest way to do it that I've seen so far.**

```
#define GPPU 0x06
#define INTF        0x07
#define INTCAP      0x08
#define GPIO        0x09
      // the main data port for reading and
      // writing
#define OLAT 0x0A
```

The pizza is the data you load into the appropriate MCP23008 register. Here's a typical write register sequence that targets the GPIO register:

```
      Wire.beginTransmission(DEVADDR);
      // transmit to motor shield
      Wire.send(GPIO);
      // select the GPIO register
      Wire.send(value);
      // data to load into the GPIO register
      Wire.endTransmission();
      // stop transmitting
```

The MotorShield library contains the following publically accessible functions:

```
void begin(void)
void writeLEDs(byte value)
byte readInputs(void)
void readButtons(bool* btn1, bool* btn2)
```

The GPIO register code snippet is actually part of the *writeLEDs* library call. To use the library functions, we must first instantiate a MotorShield class. Here is the class code that lies within the MotorShield library:

```
/*******************
 * MotorShield Class
 ******************/

class MotorShield
{
       public:
       void begin(void);
       void writeLEDs(byte value);
       byte readInputs(void);
       void readButtons(bool* btn1, bool* btn2);

       private:
       void reorder_LEDs(byte* value);
};
```

Here's how to instantiate a MotorShield class:

```
// Wire.h must be included when using the
// MotorShield.h library
#include <Wire.h>
#include <MotorShield.h>

// Create an instance of MotorShield called
// extender
MotorShield extender;
```

The next step is to finish up our variable declarations and kick off *extender*:

```
byte x;
bool btn1, btn2;

void setup()
{
    extender.begin();    // initialize the I/O
                         // extender
}
```

The *setup* function code is pretty vague. A peek under the hood will clear things up:

```
void MotorShield::begin(void)
{
       Wire.begin();
       // join the I2C bus as master

       Wire.beginTransmission(DEVADDR);
       // transmit to motor shield
       Wire.send(IODIR);
       // select the data direction register
       Wire.send(0xF0);
       // configure high half of I/O as inputs,
       // and the low half as outputs
       Wire.endTransmission();
       // stop transmitting

}
```

Setting the MCP23008's IODIR register is analogous to setting PIC I/O pins as inputs or outputs.

Let's write an application that reads the pushbuttons and illuminates the motor shield's onboard LEDs accordingly. Here's the truth table we will code by:

```
btn1 depressed - btn2 released = LED1 ON
btn1 released - btn2 depressed = LED2 ON
btn1 depressed - btn2 depressed = LED3 ON
```

I can't show you the LEDs, but I can show you the code:

```
void loop()
{
  // readButtons sets the bool values, 1 for
  // pressed button, 0 for unpressed button
  extender.readButtons(&btn1, &btn2);

  x = 0x00;

  if (btn1 && !btn2)
    x = x | 0x01;          //illuminate LED1
  if (!btn1 && btn2)
    x = x | 0x02;          //illuminate LED2
  if (btn1 && btn2)
    x = x | 0x04;          //illuminate LED3

  // writes the binary value of x (first 4 bits)
  // to the LEDs
  extender.writeLEDs(x);
}
```

Double-check my code when you get your own motor shield.

The Possibilities

You can drive a pair of DC motors, a stepper motor, and control up to four servos simultaneously. You can use the motor shield pushbuttons to activate or deactivate a motor or servo. The state of a motor, servo, or LED can be altered according to the logic levels of the jumper-switch inputs. Extend the I²C bus to control external I²C slave devices via J21. Drive small relays using the stepper motor MOSFETs. Or, just have fun writing code to make stuff move and blink LEDs. **SV**

Life with Robot!
Not a dream anymore

More than Humanoid!

HOVIS Series

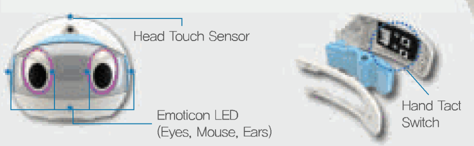
HOVIS Lite

- 18 Kinds Of Different Application Models

- 16 Kinds Of Optional Sensor Modules

HOVIS Eco Plus

- Human Robot Interaction Sesnors



Curriculum

HerkuleX Series

- Superior Resolution. Excellent Linearity. Monster Torque
- Compact Design & Easy Assembly System
- Incremental Magneto Resistant Encoder(DRS-0402/0602)

ARDUINO
ARDUINO Compatible!

	DRS-0601	DRS-0602	DRS-0401	DRS-0402	DRS-0201	DRS-0101
Size	56mm(W) X 35mm(D) X 46mm(H)		56mm(W) X 35mm(D) X 38mm(H)		45mm(W) X 24mm(D) X 32mm(H)	
Weight	145g		123g		60g	45g
Input Voltage	9.5 ~ 14.8V DC				7 ~ 12V DC	7.4V DC
Stall Torque	7.6N.m (77.0kgf.cm) at 14.8V		5.1N.m (52.0kgf.cm) at 14.8V		2.4N.m (24.0kgf.cm) at 7.4V	1.2N.m (12.0kgf.cm) at 7.4V
Max. Speed	0.164s/60˚ @14.8V		0.162s/60˚ @14.8V		0.147s/60˚ @7.4V	0.166s/60˚ @7.4V
Operating Angle	320˚, Continuous Rotation	720˚, Continuous Rotation (Built-in Speed Controller)	320˚, Continuous Rotation	720˚, Continuous Rotation (Built-in Speed Controller)	320˚, Continuous Rotation	
Resolution	2048 Steps	12960 Steps	2048 Steps	12960 Steps	1024 Steps	
Communication	Full Duplex Asynchronus Serial(TTL), Multi Drop, 0~254 ID, Max 1Mbps (0.67Mbps for DRS-0101, DRS-0201)					
Motor	Premium FAULHABER Coreless DC				Coreless DC	Metal Brush Cored DC
Position Sensor	Potentiometer	Magnetic Encoder	Potentiometer	Magnetic Encoder	Potentiometer	
Gear	Super Reinforced Metal					Engineering Plastic

Dongbu Robot Co., Ltd. www.dongburobot.com
11th Floor, Bucheon Techno Park 401, Yakdae-dong, Wonmi-gu, Bucheon-city, Gyunggi-do 420-734, Korea
TEL : +82-32-329-5551(ext.311), FAX : +82-32-329-5569, E-MAIL : robotsales@dongbu.com

Dongbu Robot

Making Better Arduino Robots with the ArdBot II

Part 4

By Gordon McComb

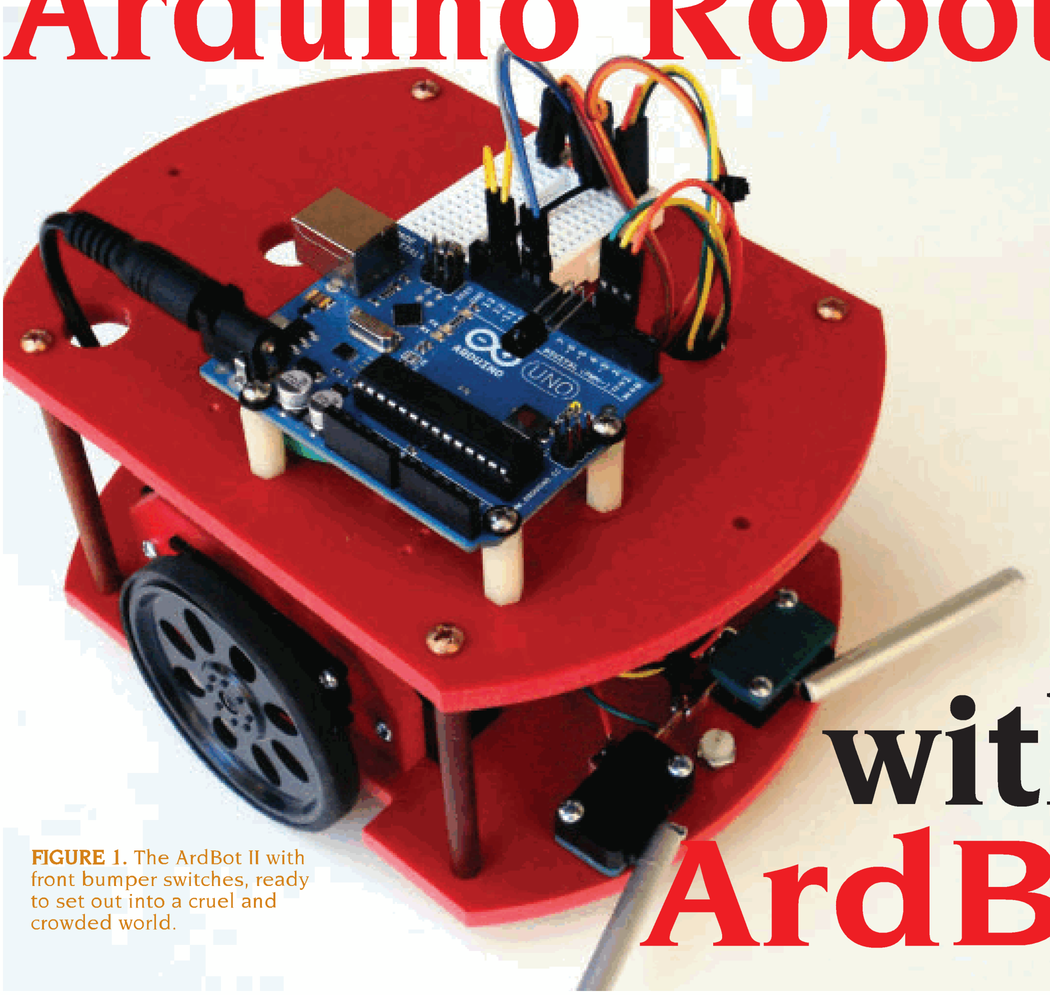


FIGURE 1. The ArdBot II with front bumper switches, ready to set out into a cruel and crowded world.

We all love a robot that does something. Even if that something is just to entertain us. The ArdBot II — an affordable and expandable robot base designed around the Arduino Uno microcontroller — both entertains and educates.

In previous installments of this series, we learned how to build the basic ArdBot II base which is shown in Figure 1. We added an Arduino controller, some motors, and couple of batteries, and out came our working bot. We also learned how to equip it with a rudimentary sense of touch, and discovered how easy it is to use simple tones to provide feedback.

This time, we take things a step further and give the ArdBot II the ability to be controlled remotely. The robot is still *autonomous* — that is, it reacts to its environment and makes cute beeping tones to let you know what it's doing. With an inexpensive infrared receiver module along with a $2 TV remote, we can also command the bot to do our bidding. After all, that's what robots are for.

In previous months, I've mentioned that certain accessories — specifically the front leaf switches and piezo speaker — are optional. In this article, I'll assume your ArdBot II includes these options.

This is Your Robot, Controlled Remotely

Remote control through an infrared light beam is surprisingly simple, thanks to low cost IR receiver modules that you can find in online electronics stores such as SparkFun and Parallax. Price varies from about $1 to $5, depending on the source. I've noted

some online destinations that offer an IR receiver module suitable for use with the ArdBot II under **Sources**. The infrared receiver module looks a bit like a fat transistor. It has three leads which can be directly inserted into the Arduino's I/O header pins as shown in **Figure 2**. Be careful: Proper orientation is a must. The module has a front side which is denoted by a "dome" or lens. I'm showing the module dome side up, with the leads bent at a 90 degree angle and inserted into pins 5, 6, and 7 of the Arduino.

You want to avoid connecting the module backwards as that might damage it. So, before connecting — especially if you use another module not indicated in the Sources box — check that the pinout connections are as follows:

Pin 1	Data output
Pin 2	Ground
Pin 3	Power

Pin 1 is the pin on the left when the dome (lens) of the module is directly facing you as shown in **Figure 3**.

With any receiver, you need a transmitter. That job falls to your basic everyday universal TV remote control. You can find these online as well, though for my robot projects I prefer the cheap units at the dollar and discount stores. As long as the remote supports the most common Sony TV codes, it'll work. (I have yet to find any universal remote control that wasn't compatible with Sony TV codes.)

Ordinarily, it's a tall job to decode the data transmitted from a remote control and intercepted by the IR receiver module. This task is made super simple, however, with a third-party Arduino code library, *IRRemote* — created by Arduino maven Ken Shirriff. This library does not come with the Arduino software and must be downloaded separately; see the Sources box for the URL. To simplify matters for you, I've included it with the download with the main Arduino sketch for this part.

To use the *IRRemote* library, you must move all its contents to your sketchbook *libraries* folder. This is a directory within the Arduino sketch folder, for example, *My Documents\Arduino\libraries*. If there isn't already a folder called *libraries*, you need to create one.

After moving the *IRRemote* library to its proper destination, you must restart your Arduino software. You must do this step or your ArdBot II sketch will not compile. The sketch will also not compile if the library is in the wrong place. So, be sure to do these things or you'll just end up getting frustrated.

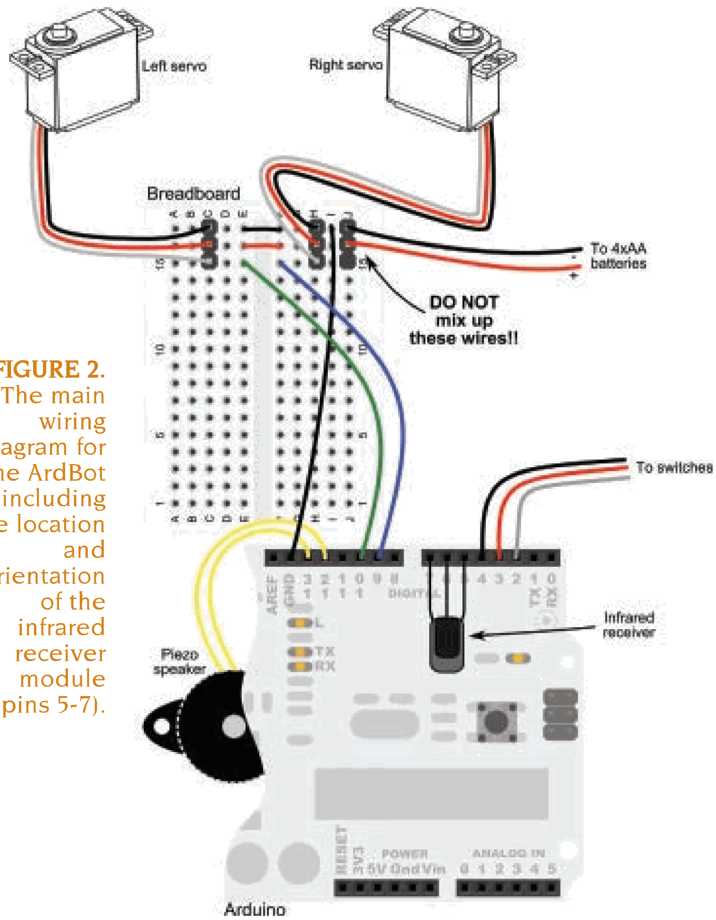


FIGURE 2. The main wiring diagram for the ArdBot II, including the location and orientation of the infrared receiver module (pins 5-7).

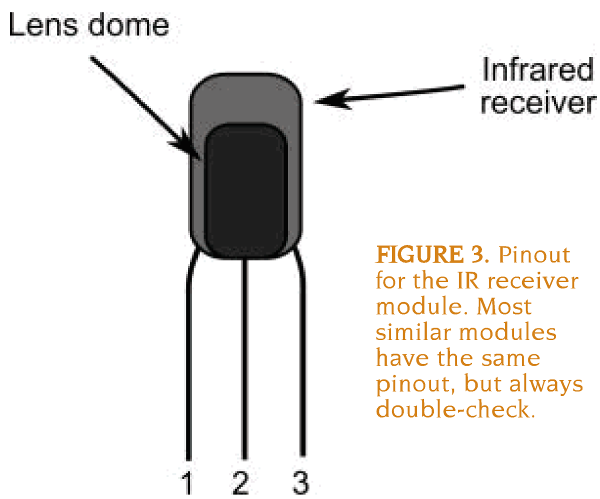


FIGURE 3. Pinout for the IR receiver module. Most similar modules have the same pinout, but always double-check.

Refer to Part 1 of this series for a full list of mechanical parts for the ArdBot II.

The IR Remote library is available from the GitHub repository. Be sure you are using the version for the Arduino IDE 1.0 software.

github.com/shirriff/Arduino-IRremote

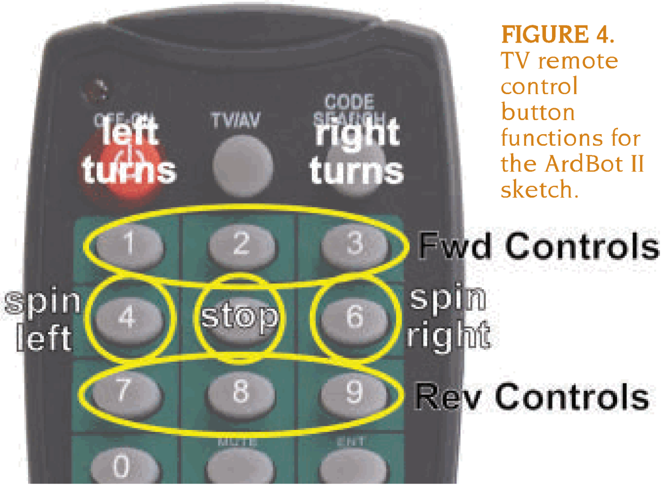


FIGURE 4. TV remote control button functions for the ArdBot II sketch.

Listing 1 shows the full text of the ArdBot II demonstration code. Note that in addition to the *IRRemote* library, the sketch relies on the built-in servo library (part of the Arduino software), as well as a definitions file called *pitches.h*. We introduced this file and how it's used in last month's installment. To control the robot using a universal TV remote control, you must first set the remote to use a compatible Sony TV code. Follow the instructions packed with your remote and test the various code settings until you find one that works.

Figure 4 shows how the buttons on the remote are mapped to various control procedures in the ArdBot II sketch. For example, pressing the ***2*** button makes the robot move straight forward. Pressing ***5*** stops it. Pressing ***4*** makes

LISTING 1. *ArdBotII.*

```
/*
  Be sure to move the IRremote folder contained
  with this sketch to the Arduino sketchbook
  libraries folder. See the readme.txt file for
  more information.

  Program your universal remote for a Sony TV
  code.
  Try different codes until you get one that
  works.

*/

#include <IRremote.h>
#include <Servo.h>
#include "pitches.h"

Servo servoLeft;      // Define left servo
Servo servoRight;     // Define right servo

int RECV_PIN = 5;
IRrecv irrecv(RECV_PIN);
decode_results results;
volatile int pbLeft = LOW;
volatile int pbRight = LOW;
boolean started = false;

void setup() {
  // Set pin modes for switches
  pinMode(2, INPUT);
  pinMode(3, INPUT);
  pinMode(4, OUTPUT);
  digitalWrite(2, HIGH);
  digitalWrite(3, HIGH);
  digitalWrite(4, LOW);
       // Serves as ground connection

  pinMode(6, OUTPUT);
       // IR power, ground pins
  pinMode(7, OUTPUT);
  digitalWrite(6, LOW);       // IR ground
  digitalWrite(7, HIGH);      // IR power

  pinMode(12, OUTPUT);        // Ground for
                              // speaker
  digitalWrite(12, LOW);

  servoLeft.attach(10);
       // Set left servo to digital pin 10
  servoRight.attach(9);
       // Set right servo to digital pin 9
  irrecv.enableIRIn();
       // Start the receiver
  Serial.begin(9600);

  // Set up interrupts
  attachInterrupt(0, hitRight, FALLING);
  attachInterrupt(1, hitLeft, FALLING);

  started = true;
}

void loop() {

  if (pbLeft == HIGH) {
       // If left bumper hit
    int tones[] = {NOTE_C4, NOTE_B3, NOTE_C4};
    int toneDurations[] = {4,4,4};
    reverse();
    makeTone(tones, toneDurations, sizeof
    (tones)/sizeof(int));
    delay(500);
    spinRight();
    delay(1500);
    forward();
    pbLeft = LOW;
    Serial.println("pbLeft");
  }

  if (pbRight == HIGH) {
       // If right bumper hit
    int tones[] = {NOTE_D6, NOTE_C3};
    int toneDurations[] = {4,4};
    reverse();
    makeTone(tones, toneDurations,
sizeof(tones)/sizeof(int));
    delay(500);
    spinLeft();
    delay(1500);
    forward();
    pbRight = LOW;
    Serial.println("pbRight");
  }

  if (irrecv.decode(&results)) {
    switch (results.value) {
      case 0x10:
        Serial.println("1");
       // Turn left forward
        turnLeftFwd();
        break;
      case 0x810:
        Serial.println("2");
       // Forward
        forward();
        break;
      case 0x410:
        Serial.println("3");
```

it spin left, and so on. Note there are two kinds of turns: spin and slip (or normal) turns.

- In a spin turn, the two wheels of the robot counter rotate, causing it to spin in place — thus the name "spin turn."
- In a normal or slip turn, one wheel turns while the other stops. This causes the ArdBot II to turn in a more gradual arc.

Though you can operate the robot with the remote control, it is still reactive if it bumps into an object. If one of its front bumper switches strikes something, that'll cause the robot to back up, turn, and head off into a new direction.

This process and the programming for it were detailed in Part 2 of my previous series, *Making Arduino Robots with the ArdBot II*.

Inside the ArdBot II Sketch

The *ArdBotII.ino* sketch in **Listing 1** builds upon the code we've developed previously. In this version of the sketch, routines are added to listen for coded signals from the infrared remote control. Upon receiving one of these signals, the robot acts accordingly, such as stopping, moving backwards, or turning.

Here's a more detailed rundown of the main parts of the ArdBot II sketch:

```
        // Turn right forward
          turnRightFwd();
          break;
        case 0xC10:
          Serial.println("4");
        // Spin left
          spinLeft();
          break;
        case 0x210:
          Serial.println("5");
        // Stop
          stopRobot();
          break;
        case 0xA10:
          Serial.println("6");
        // Spin right
          spinRight();
          break;
       case 0x610:
          Serial.println("7");
        // Turn left reverse
          turnLeftRev();
          break;
        case 0xE10:
          Serial.println("8");
        // Reverse
          reverse();
          break;
        case 0x110:
          Serial.println("9");
        // Turn right reverse
          turnRightRev();
          break;
      }
      //Serial.println(results.value, HEX);
      irrecv.resume(); // Receive the next value
      delay(2);
   }
}

void makeTone(int tones[], int toneDurations[],
int length) {
   // Iterate notes of tune
   for (int thisNote = 0; thisNote < length;
   thisNote++) {

     //Calculate the note duration
     int toneDuration =
1000/toneDurations[thisNote];
     tone(13, tones[thisNote],toneDuration);

     //Add slight pause between notes
     int pauseBetweenNotes = toneDuration *
     1.30;
     delay(toneDuration * 1.30);
     noTone(13);                 // Stop tone
  }
  irrecv.enableIRIn();       // Restart receiver
  return;
}

// Motion routines for forward, reverse, turns,
// and stop
void forward() {
  servoLeft.write(180);
  servoRight.write(0);
}
void reverse() {
  servoLeft.write(0);
  servoRight.write(180);
}
void spinLeft() {
  servoLeft.write(0);
  servoRight.write(0);
}
void spinRight() {
  servoLeft.write(180);
  servoRight.write(180);
}
void turnLeftFwd() {
  servoLeft.write(90);
  servoRight.write(0);
}
void turnRightFwd() {
  servoLeft.write(180);
  servoRight.write(90);
}
void turnLeftRev() {
  servoLeft.write(90);
  servoRight.write(180);
}
void turnRightRev() {
  servoLeft.write(0);
  servoRight.write(90);
}
void stopRobot() {
  servoLeft.write(90);
  servoRight.write(90);
}

// Interrupt handlers
void hitLeft() {
  if (started)
    pbLeft = HIGH;
}
void hitRight() {
  if (started)
    pbRight = HIGH;
}
```

```
#include <IRremote.h>
#include <Servo.h>
#include "pitches.h"
```

These lines of code insert ("include") three additional files which provide extra features:

- *IRremote* is a third-party *object library* that must be added to the */libraries* directory in your Arduino sketchbook folder.
- *Servo* is an object library that's included with the Arduino environment.
- *pitches.h* is an extra file of pre-defined numeric *constants*; these constants make it easier to specify a note to play through the speaker. The *pitches.h* file is located with the sketch.

```
Servo servoLeft;
Servo servoRight;
```

These two lines are *object constructors*; they define two *Servo* objects that represent the physical servos on the robot. The names of the objects are descriptive to help you follow what they are for.

```
int RECV_PIN = 5;
IRrecv irrecv(RECV_PIN);
decode_results results;
```

These lines set up the infrared receiver. Note that the receiver input is connected to pin D5.

```
volatile int pbLeft = LOW;
volatile int pbRight = LOW;
boolean started = false;
```

Each of these lines above define a *variable* used elsewhere in the sketch. Variables are temporary holders of data that can be re-used. The *int* and *boolean* keywords specify the kind of variable; in this case, a 16-bit (two byte) integer (whole number) and a true/false boolean.

```
void setup() {
```

Every Arduino sketch must contain a *setup* function. This is where any programming statements go that literally set up the Arduino before things get going. These statements are executed only once.

```
pinMode(2, INPUT);
pinMode(3, INPUT);
pinMode(4, OUTPUT);
digitalWrite(2, HIGH);
digitalWrite(3, HIGH);
digitalWrite(4, LOW);
```

The two bumper switches on the front of the robot are connected to Arduino pins D2, D3, and D4. These lines use the *pinMode* statement to specify pins D2 and D3 are inputs, and D4 is an output. They then use the *digitalWrite* statement to indicate whether the pins are LOW (0V) or HIGH (5V).

Though pins D2 and D3 are inputs, they're made HIGH to set their internal pull-up resistor. This is necessary for proper operation of the switches.

Finally, pin D4 is made LOW so that it acts as a 0V reference for the switches.

```
pinMode(6, OUTPUT);
pinMode(7, OUTPUT);
digitalWrite(6, LOW);
digitalWrite(7, HIGH);
```

Here, more pins are set as outputs. Pin D6 is made LOW to act as 0V ground; pin D7 is made HIGH to act as 5V power (this method is acceptable as long as the device being powered does not draw more than 25-30 mA of current).

```
pinMode(12, OUTPUT);
digitalWrite(12, LOW);
```

These lines define pin D12 as a 0V ground connection to the piezo speaker. The other lead of the piezo speaker connects to pin D13.

```
servoLeft.attach(10);
servoRight.attach(9);
```

These two lines of code electrically attach the two servos to pins D10 and D9.

```
irrecv.enableIRIn();
Serial.begin(9600);
```

These lines of code start the infrared receiver and begin serial connection back to the PC. Messages sent from the Arduino are received into the Serial Monitor window.

```
attachInterrupt(0, hitRight, FALLING);
attachInterrupt(1, hitLeft, FALLING);
```

The bumper switches use an Arduino feature known as *hardware interrupts* to determine when the switches are activated. Interrupts literally interrupt the action of the sketch to take time out to run some specified piece of code:

- *0* and *1* specify the interrupt number (0 = pin D2; 1 = pin D3).
- *hitRight* and *hitLeft* are the sketch functions that are called when an interrupt occurs.
- *FALLING* is a directive that tells the Arduino to trigger the interrupt when the signal to pin D2 or D3 falls; that is, changes from HIGH to LOW.

```
void loop() {
```

Every Arduino sketch must include a *loop* function. Inside this function are the programming statements that repeat over and over again as the robot operates.

```
if (pbLeft == HIGH) {
  int tones[] = {NOTE_C4, NOTE_B3, NOTE_C4};
  int toneDurations[] = {4,4,4};
  reverse();
  makeTone(tones, toneDurations,
  sizeof(tones)/sizeof(int));
  delay(500);
  spinRight();
  delay(1500);
  forward();
  pbLeft = LOW;
  Serial.println("pbLeft");
}
```

This *if* test structure commands the Arduino to go through a series of steps if the left bumper switch is activated. The code first defines some musical notes to play and sets the robot in reverse. After a short 1/2 second delay — *using delay(500)* — the robot spins right, waits another 1.5 seconds, then goes forward again.

A separate *if* test determines when the right bumper switch is activated.

```
if (irrecv.decode(&results)) {
  switch (results.value) {
    case 0x10:
      Serial.println("1");
      turnLeftFwd();
      break;
```

These lines of code check to see if there's any recent commands sent from the infrared remote control. If there are, the sketch matches the command value received, then operates the robot accordingly. For example, if the code is hexadecimal 10 (shown as *0x10*), then the robot turns left while going forward. As a visual check, if the robot is attached to your PC via the USB cable, the code also displays a "1" — for button 1 — in the Serial Monitor window. Buttons 2 through 9 are likewise matched; each button makes the robot do something different.

```
void makeTone(int tones[], int
toneDurations[], int length) {
...
```

makeTone is a user-defined function. It sounds one or more tones through the piezo speaker attached to pins 12 and 13. The code is a bit elaborate, and more than can be described here. For a detailed description, see the Tone example provided on the *arduino.cc* website.

```
void forward() {
  servoLeft.write(180);
  servoRight.write(0);
}
```

The *forward* function is one of several motion control routines that operate the two servos to move the robot. Each servo is command to a specific position; either 0, 90, or 180:

- *0* makes the servo turn continuously in one direction.
- *90* makes the servo stop.
- *180* makes the servo turn continuously in the other direction.

```
void hitLeft() {
  if (started)
    pbLeft = HIGH;
}
```

The *hitLeft* function is automatically called when the left bumper is activated. The function sets the variable *pbLeft* to HIGH. This variable is checked each time the Arduino goes through the loop function. A separate function *hitRight* is for the right bumper.

Coming Up: Arduino and Sensor Alternatives

In the next and final installment, we'll look at some alternatives to the Arduino Uno microcontroller board, as well as explore using all-in-one shields that incorporate servo headers and solderless breadboards — they make wiring things up very easy. We'll also discuss some other low cost sensors you can connect to your ArdBot II to give it a greater sense of the wide wide world around it. **SV**

Sources

Budget Robotics
Precut ArdBot II body chassis with all assembly hardware
www.budgetrobotics.com

Selected sources for Arduino and other parts. For the infrared receiver, consider getting two so you have one for a spare:

All Electronics
www.allelectronics.com

BG Micro
www.bgmicro.com

Jameco
www.jameco.com

Mouser
Infrared receiver: 782-TSOP38238
www.mouser.com

SparkFun
Infrared receiver: SEN-10266
www.sparkfun.com

Parallax
Infrared receiver: 350-00014
www.parallax.com

Pololu
www.pololu.com

Lynxmotion

Imagine it. Build it. Control it.™

SERVO ERECTOR SET V1.1

FLOWBOTICS STUDIO

Introducing the Lynxmotion Servo Erector Set (SES) V1.1 with the new FlowBotics Studio V2. The SES V1.1 includes over 500 parts which are used to build and program a variety of different robots including a wheeled rover, biped, quadruped, hexapod and robotic arm. FlowBotics Studio is a powerful, versatile and easy to use graphical software suite which is compatible with a variety of robots and parts. FlowBotics Studio includes the complete FlowStone programming language!

Available now at www.Lynxmotion.com and www.RobotShop.com

Lynxmotion and FlowBotics Studio are trademarks of RobotShop Distribution Inc.

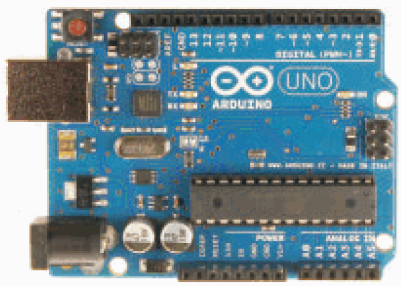
RobotShop.com
Putting robotics at your service™

CanaKit™
Electronic Kits • Modules • Parts

**Save 10% using Coupon Code SM2012*
when ordering online at www.canakit.com**

Motor Controllers • Arduino • Arduino Kits • Arduino Shields • Robot Platforms • Parts

Arduino Uno



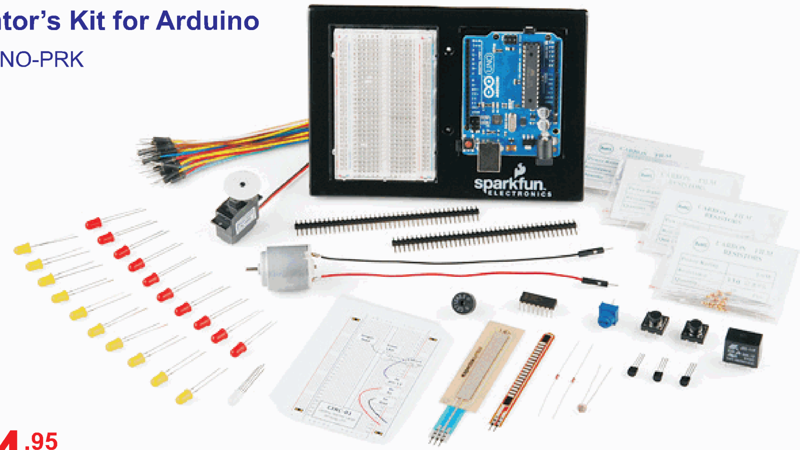
$29.95

ARDUINO-UNO

Latest "R3" Version!

Inventor's Kit for Arduino

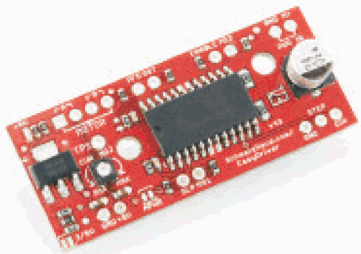
ARDUINO-PRK



$94.95

Includes 36-Page Printed Full-Color Manual

EasyDriver Stepper Motor Driver



SX09402

$14.95

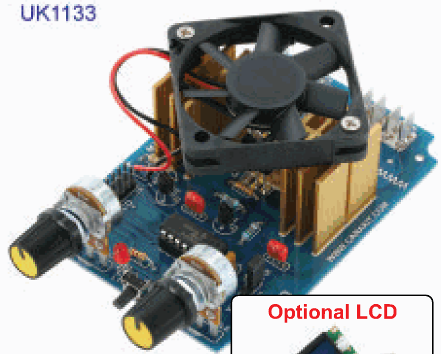
Stepper Motor

SX09238

$19.95

50A Motor Speed Controller (PWM)

UK1133



$59.95

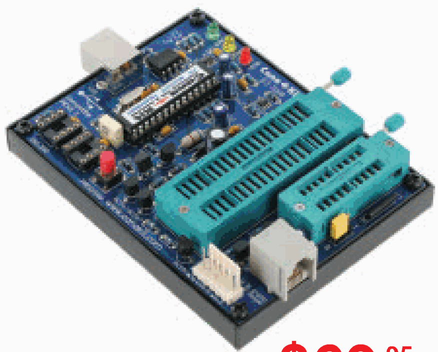
Optional LCD

7A Bi-Directional Motor Speed Controller (PWM)

UK1125

$39.95

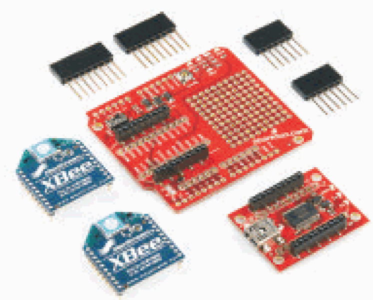
USB PIC Programmer (MPLAB Compatible)



UK1300

$69.95

Xbee Wireless Kit



XBEE-KIT

$94.95

1•888•540-KITS (5487)   info@canakit.com   **www.canakit.com**

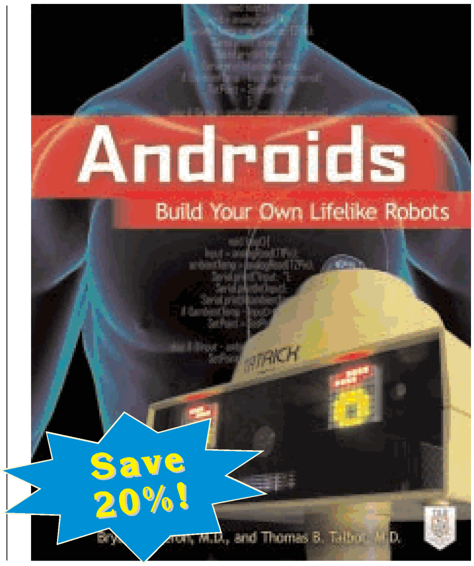
* Limit one per customer. Can not be combined with other coupons or volume discounts. Does not apply to products already on sale. Prices subject to change without notice.

The SERVO Webstore

Attention Subscribers ask about your discount on prices marked with an *

CD-ROM SPECIALS

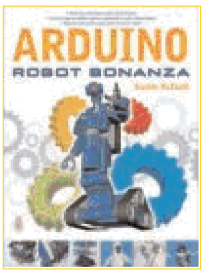
11.2013 FEATURED



ROBOTICS

Arduino Robot Bonanza

by Gordon McComb



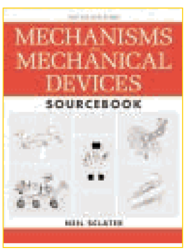
Create high-tech walking, talking, and thinking robots. "McComb hasn't missed a beat. It's an absolute winner!"

Breathe life into the robots of your dreams — without advanced electronics or programming skills. *Arduino Robot Bonanza* shows you how to build autonomous robots using ordinary tools and common parts. Learn how to wire things up, program your robot's brain, and add your own unique flair.

$29.95*

Mechanisms and Mechanical Devices Sourcebook 5th Edition

by Neil Sclater



Fully revised throughout, this abundantly illustrated reference describes proven mechanisms and mechanical devices. Each illustration represents a design concept that can easily be recycled for use in new or modified mechanical, electromechanical, or mechatronic products. Tutorials on the basics of mechanisms and motion control systems introduce you to those subjects or act as a refresher.

Reg $89.95

Making Things Move: DIY Mechanisms for Inventors, Hobbyists, and Artists

by Dustyn Roberts

In *Making Things Move: DIY Mechanisms for Inventors, Hobbyists, and Artists*, you'll learn how to successfully build moving mechanisms through non-technical explanations, examples, and do-it-yourself projects — from kinetic art installations to creative toys to energy-harvesting devices. Photographs, illustrations, screenshots, and images of 3D models are included for each project.

$29.95*

Build Your Own Humanoid Robots

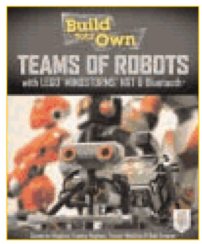
by Karl Williams

GREAT 'DROIDS, INDEED!

This unique guide to sophisticated robotics projects brings humanoid robot construction home to the hobbyist. Written by a well-known figure in the robotics community, *Build Your Own Humanoid Robots* provides step-by-step directions for six exciting projects, each costing less than $300. Together, they form the essential ingredients for making your own humanoid robot.

$24.95*

NEW! Build Your Own Teams of Robots with LEGO® MINDSTORMS® NXT and Bluetooth®

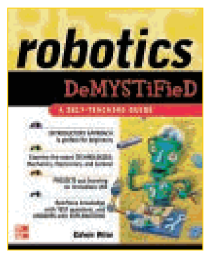


Plan, design, assemble, and program robot squads that communicate and cooperate with each other to accomplish together what they can't do individually. Inside the authors show you how to construct a team capability matrix (TCM) and use the Bluetooth Robotic-Oriented Network (BRON) so your robot teams can share sensors, actuators, end effectors, motor power, and programs. **$30.00**

Robotics Demystified

by Edwin Wise

YOU DON'T NEED ARTIFICIAL INTELLIGENCE TO LEARN ROBOTICS!



Now anyone with an interest in robotics can gain a deeper understanding — without formal training, unlimited time, or a genius IQ. In *Robotics Demystified*, expert robot builder and author Edwin Wise provides an effective and totally painless way to learn about the technologies used to build robots! **$19.95**

We accept VISA, MC, AMEX, and DISCOVER
Prices do not include shipping and may be subject to change.

To order call 1-800-783-4624

SERVO Magazine Bundles

Now you can get one year's worth of all your favorite articles from *SERVO Magazine* in a convenient bundle of print copies. Available for years 04, 05, 06, 07, 08, 09, 10, 11 and 2012.

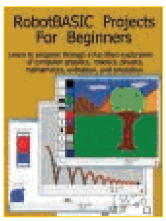
Any bot builders out there? Get cool robotics stuff from my store!

Call me at my order desk!

Visit my online store @ www.servomagazine.com

RobotBASIC Projects For Beginners

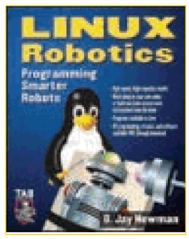
by John Blankenship, Samuel Mishal



If you want to learn how to program, this is the book for you. Most texts on programming offer dry, boring examples that are difficult to follow. In this book, a wide variety of interesting and relevant subjects are explored using a problem-solving methodology that develops logical thinking skills while making learning fun. RobotBASIC is an easy-to-use computer language available for any Windows-based PC and is used throughout the text.
Price $14.95

Linux Robotics

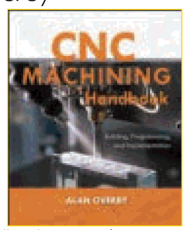
by D. Jay Newman



If you want your robot to have more brains than microcontrollers can deliver — if you want a truly intelligent, high-capability robot — everything you need is right here. *Linux Robotics* gives you step-by-step directions for "Zeppo," a super-smart, single-board-powered robot that can be built by any hobbyist. You also get complete instructions for incorporating Linux single boards into your own unique robotic designs. No programming experience is required. This book includes access to all the downloadable programs you need.
$38.95

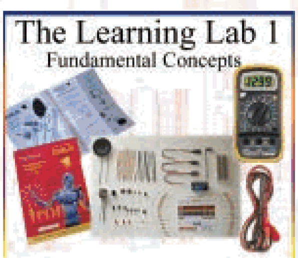
CNC Machining Handbook: Building, Programming, and Implementation

by Alan Overby

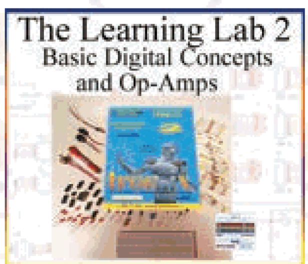


The *CNC Machining Handbook* describes the steps involved in building a CNC machine and successfully implementing it in a real world application. Helpful photos and illustrations are featured throughout. Whether you're a student, hobbyist, or business owner looking to move from a manual manufacturing process to the accuracy and repeatability of what CNC has to offer, you'll benefit from the in-depth information in this comprehensive resource. **$34.95**

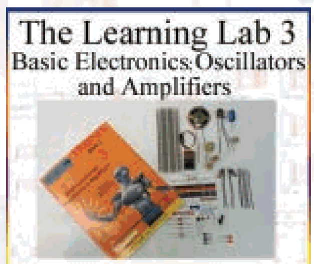
FOR BEGINNER BOT BUILDERS



$59.95



$49.95



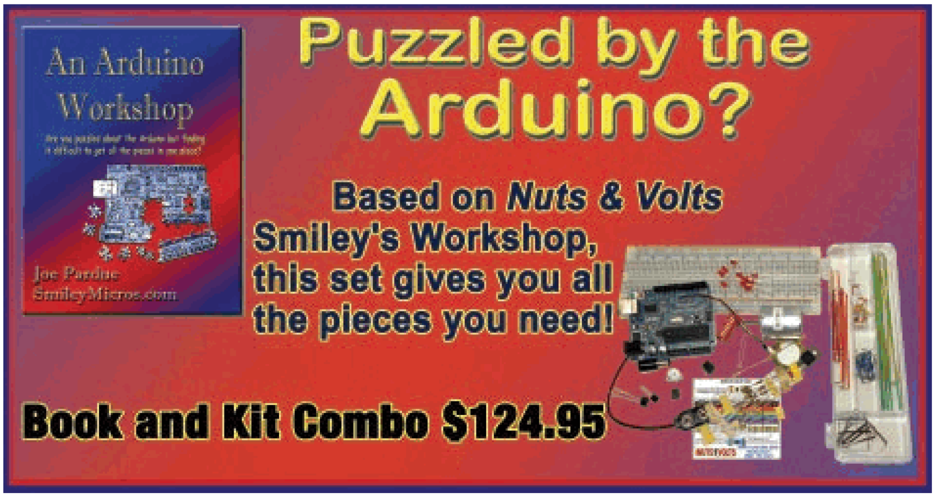
$39.95

The labs in this series — from GSS Tech Ed — show simple and interesting experiments and lessons, all done on a solderless circuit board. As you do each experiment, you learn how basic components work in a circuit, and continue to build your arsenal of knowledge with each successive experiment.

For more info and a promotional video, please visit our webstore.

Or order online www.servomagazine.com

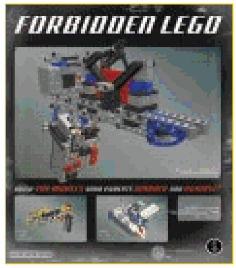
SPECIAL OFFERS



Forbidden LEGO

by Ulrik Pilegaard / Mike Dooley

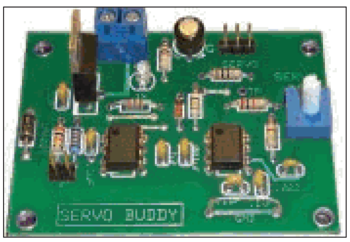
Forbidden LEGO introduces you to the type of free-style building that LEGO's master builders do for fun in the back room. Using LEGO bricks in combination with common household materials (from rubber bands and glue to plastic spoons and ping-pong balls) along with some very unorthodox building techniques, you'll learn to create working models that LEGO would never endorse.



Reg $24.95 Sale Price $19.95

PROJECTS

The SERVO Buddy Kit



An inexpensive circuit you can build to control a servo without a microcontroller.

For more information, please check out the May 2008 issue or go to the SERVO webstore.

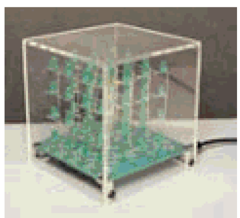
Includes an article reprint.

Subscriber's Price **$39.55**

Non-Subscriber's Price **$43.95**

3D LED Cube Kit

From the article "Build the 3D LED Matrix Cube" as seen in the August 2011 issue of *Nuts & Volts Magazine*.

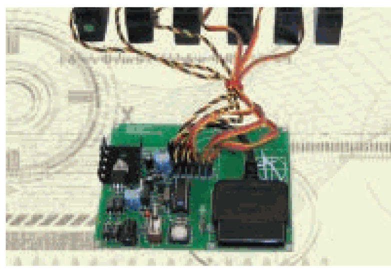


This kit shows you how to build a really cool 3D cube with a 4 x 4 x 4 monochromatic LED matrix which has a total of 64 LEDs. The preprogrammed microcontroller that includes 29 patterns that will automatically play with a runtime of approximately 6-1/2 minutes.
Colors available: Green, Red, Yellow & Blue.
Jig and plastic cases also available.

Subscriber's Price **$57.95**

Non-Subscriber's Price **$59.95**

PS2 Servomotor Controller Kit

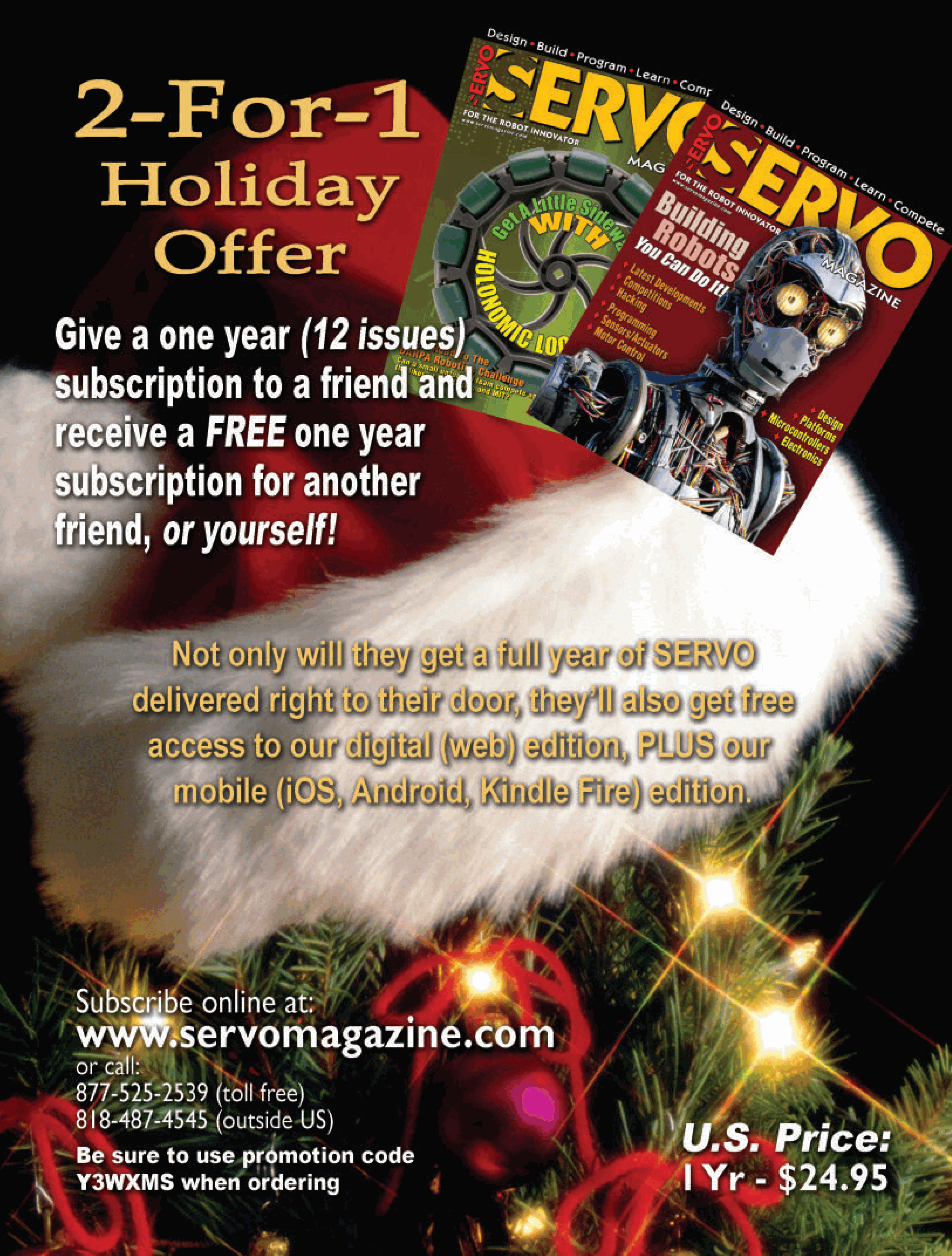


This kit accompanied with your own PlayStation controller will allow you to control up to six servomotors. Includes all components and instruction manual.

For more information, please see the February 2011 edition of *SERVO Magazine*. Assembled units available!

Subscriber's Price **$79.95**

Non-Subscriber's Price **$84.95**


2-For-1
Holiday
Offer
Give a one year (12 issues) subscription to a friend and receive a FREE one year subscription for another friend, or yourself!
Not only will they get a full year of SERVO delivered right to their door, they'll also get free access to our digital (web) edition, PLUS our mobile (iOS, Android, Kindle Fire) edition.
Subscribe online at:
www.servomagazine.com
or call:
877-525-2539 (toll free)
818-487-4545 (outside US)
Be sure to use promotion code Y3WXMS when ordering
U.S. Price:
1 Yr - $24.95
Design • Build • Program • Learn • Compete
SERVO
FOR THE ROBOT INNOVATOR
MAGAZINE
Building Robots You Can Do It!
Latest Developments
Competitions
Hacking
Programming
Sensors/Actuators
Motor Control
Design
Platforms
Microcontrollers
Electronics
HOLONOMIC LOC

Eye-Ears
My Interactive Looking/Hearing Robot

by Alan Schilling

Post comments on this article and find any associated files and/or downloads at **www.servomagazine.com/index.php?/magazine/article/november2013_Schilling.**

Yes, Eye-Ears is a little odd looking, but he is uniquely entertaining. The LED "eyes" wander back and forth, looking around the room. When someone approaches from either side of the ultrasound sensors (ears), it causes the eyes to jump and turn in their direction. Adults are surprised and children squeal with delight as Eye-Ears responds to their movements.

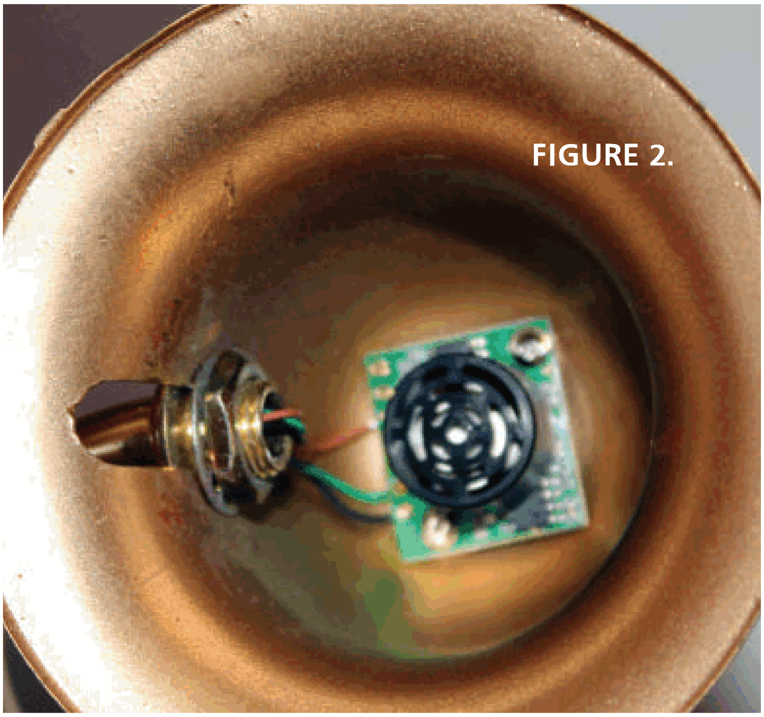


FIGURE 2.

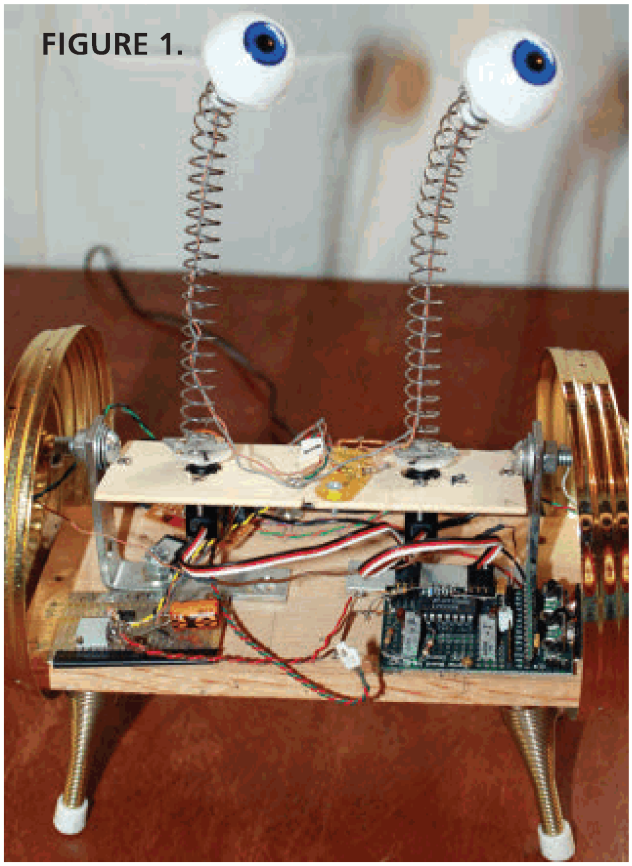


FIGURE 1.

Eye-Ears Interactive Design

A Parallax BASIC Stamp 1 microcontroller is the heart of this project. The eyes are controlled by two servos that normally rotate back and forth. When the ultrasound ears are activated, the program jumps to rotate the eyes left or right, depending on the input direction.

Description

Figure 1 shows Eye-Ears with the cover removed. Plastic toy eyes are mounted at the end of bent springs that are about 5" long. The eyes have white LEDs mounted inside which can be seen glowing from across a room.

The other end to the eye spring is attached to a set of servos (left/right) that cause them to rotate as controlled from the Stamp program.

A sonar range finder (LV-MaxSonar-EZ1) is mounted inside each ear. **Figure 2** shows the sonar range finder mounted in one of the ears (old copper cups). The analog output from the sonar goes through a comparator, then to the input of the Stamp. The PWM output of the Stamp controls the servo's movement to the eyes.

Parts List

Description	Type	Value	Source
LM339	IC, Quad Comparator		**AllElectronics.com**
R1,R2	Resistor	1K ohm, 1/4 watt	**Jameco.com**
R3	Resistor	470 ohm, 1/4 watt	**Goldmine-elec.com**
LED1,LED2	Small LEDs		**RadioShack**
LED L,R	Eye LEDs, super bright white (left/right)		
LED3	Flashing Rainbow LED		
LM7805	Regulator	Five volt	
LM7806	Regulator	Six volt	
C1,C2	Capacitor (filter)	470 mfd @ 50volts	
P1,P2	Potentiometers	10K, 10 turn	
Switch	Mini toggle SPST (power on/off)		
Transforrmer	Wall transformer	Nine volts	
Miscellaneous Parts (Artistic)			
Toy Eyeballs			
Main body (enclosure) brass/copper metals			Hobby Lobby
Legs	Door Stops		Hardware store
Springs	Eyeball extenders (about five inches)		Hardware store
Ears	Cups, brass, etc.		Garage sales
Light Fixtures			Antique Shops, etc.
BASIC Stamp 1 (or 2) microcontroller			**Parallax.com**
Stamp 1 carrier board (with nine volt connector)			
Stamp 1 serial adapter (you will also need to be able to use the editor and connection cable) or new BASIC Stamp 2 Activity Kit, USB version			
Servos 1 & 2	Cirrus CS-20BB (micro servos) (left & right)		
Sonar 1 & 2	MaxSonar-EZ -EZ1		**Maxbotix.com**

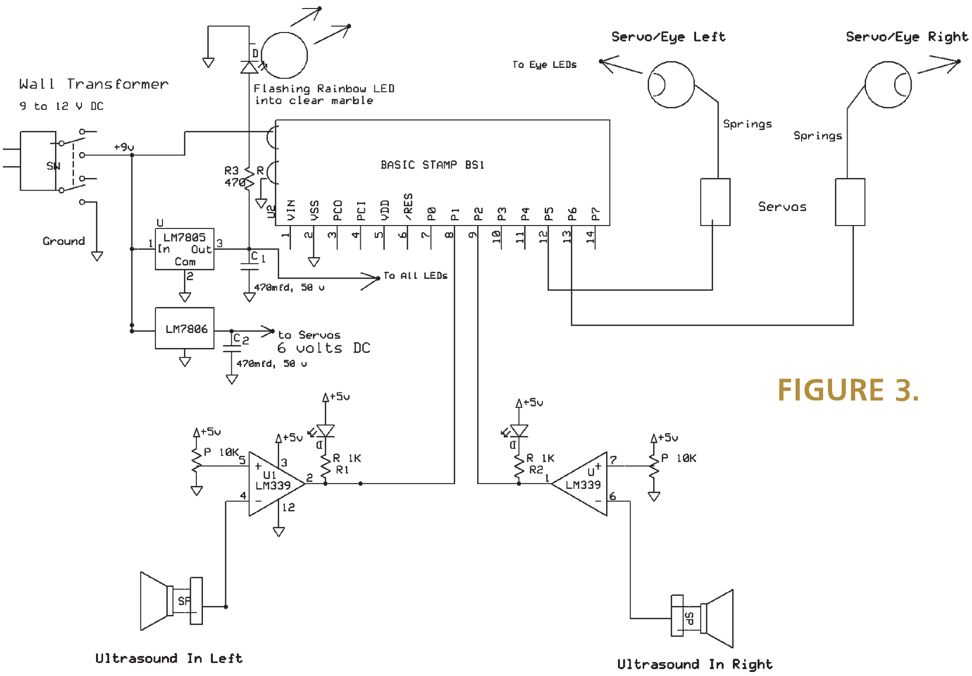


FIGURE 3.

rejection. The reference voltage on pins 5 and 7 are set to about 0.4 volts — just high enough to prevent false triggering on background noise. Normally (no signal), the comparator outputs (ports 1 and 2) are low (LEDs on).

When the sonar signal exceeds the threshold, the output goes high which then goes to port 1 or 2 on the microcontroller. A Pulse Width Modulation (PMW) signal then goes to drive the eye servos to control their position. The power supply wires that go to the servos and sonar are not shown.

Schematic Description

Figure 3 is the schematic diagram for Eye-Ear. Main power is derived from a nine volt, 500 mA wall transformer. Nine volts from the transformer goes directly to the carrier board's nine volt battery connector. Voltage is regulated down with a LM7805; five volts for U1, all LEDs, and a LM7806; six volts for the servos. C1 and C2 are electrolytic capacitors to supply surge power during servo operation.

I used the analog output from the MaxSonar units that goes to five volts when objects are at close range. The sonar signal goes through an LM339 comparator for noise

Software Program

The Stamp 1 is used for the program written in PBasic. The basic idea of the program is to rotate the eyes (servos) back and forth about 180 degrees, while checking for sonar signal inputs. A flow diagram of the program can be seen in **Figure 4**. The PULSOUT (port#) pulse time is used to control the servo position.

The Stamp controls the position of the servos from 100 (max CW) to 150 (center), then 200 (max CCW).

(For BASIC Stamp 2s, the pulse time may be 500, 750, and 1,000 for similar positions).

My program starts with the counter set to 150 (center), then the counter counts down (minus two count) to 100 at the position of max CW. The *IF* statement checks the count each time through the loop.

When the count is equal to or less than 100, the program jumps to the up count (plus two) branch, and the servo goes to the 200 count, CCW. Finally, there is a "Check Input" subroutine where the input from the sonar is tested each time through the loop. The program jumps to the right or the left, depending on which input is detected (high).

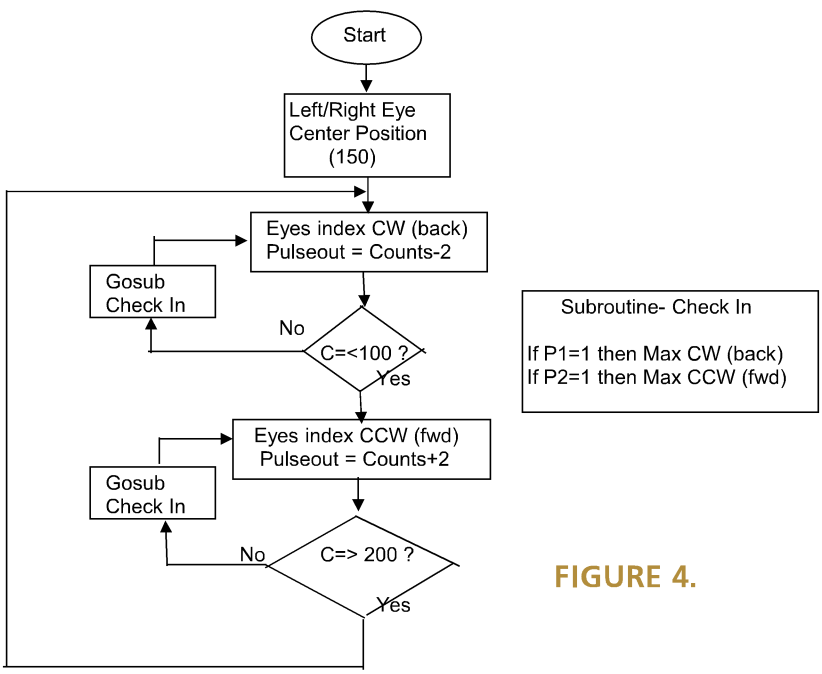


FIGURE 4.

Construction

This project (as are many of mine) is of an artistic nature and does not conform to any strict pre-arranged plans. As it goes, I have boxes full of parts; I have an idea of what I want to do; and I just have fun fitting things together to achieve my goals.

If you look at the photos, you will see that Eye-Ears is made of some common household

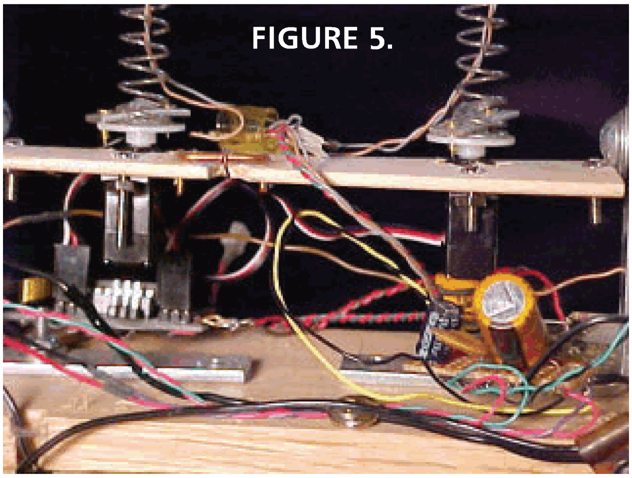


FIGURE 5.

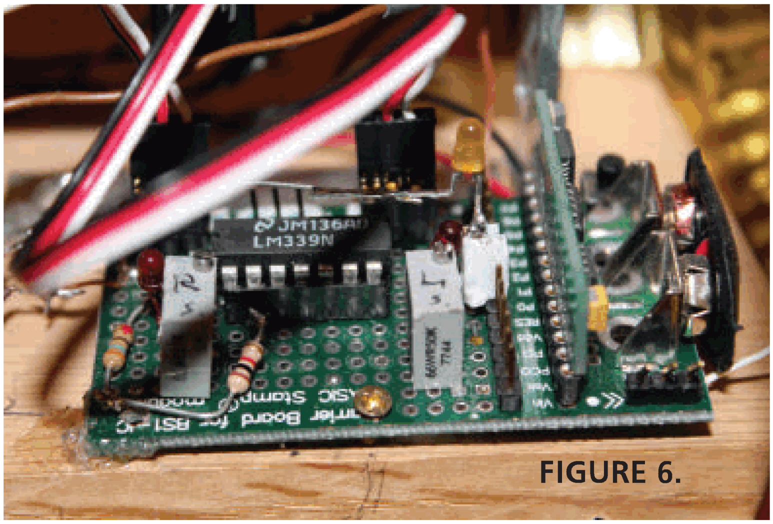


FIGURE 6.

items such as lamp fixture parts (ends), legs (spring doorstops), and ears (copper cups).

The wooden base is 3-1/2 x 7 inches, and was made to fit into the lamp base near the bottom. The door stops are screwed into the bottom wood, and the copper cups had holes drilled to fit the lamp fixture. There is a 2 x 5 inch platform mounted about two inches above the base on which the two eye servos are mounted (see **Figure 5**).

In my first design, I had a third servo mounted to the base with a rod to control the platform tilting to swivel both eyes forward and backward. The tilting moved the platform and eyes outside the exit hole range and also the mechanics jammed, so I had to abandon this idea.

The Stamp carrier board is mounted on the lower right side (see **Figure 6**). The microcontroller board is plugged in vertically; you can see board numbers by the side of the board. All of the electronic components are mounted on the Stamp carrier board.There is just one IC (LM339), a couple of potentiometers, resistors, and three LEDs. The regulator ICs and capacitors are mounted on two different small boards in the front left and back.

To add a little extra pizzaz, I mounted a flashing rainbow LED to the front of Eye-Ears. I have found that by inserting a clear glass marble in front of the rainbow LED it appears to magnify the effects of the display several fold, thus improving the color presentation.

Electronic Assembly

All of the electronics are hard-wired to the Stamp carrier printed circuit board (PCB). Do not install the microcontroller onto the carrier board until after your soldering is done on the Stamp board. Install the 16-pin IC socket first, then the two potentiometers, and finally the output resistors and LEDs. I used a nine volt battery connector to run the power supply directly into the carrier board. Be sure polarity is correct.

Standard 2 mm pitch headers are used to connect directly to each servo. The six volt power supply will need to be wired directly to the headers for servo power. Many servos have different connections as far as which input is the power (+ and -) and servo signal. You may have to experiment somewhat to wire the header correctly. (On my small servo — Cirrus CS-20 BB — the red wire is +; the brown wire is -; and the orange wire is the PWM signal from the microcontroller.

Be sure to tie the grounds from the five volt regulator and the six volt and nine volts together, so there is a common ground. The Max Sonar requires five volts regulated, and needs three wires for +/- power and a signal output lead.

Before turning on power, check for shorts by measuring with an ohmmeter between pins 1 and 2 of the Stamp, and pins 3 and 12 of the IC socket. Before you plug in the ICs, connect power and check for proper voltage at the IC sockets (five volts), the servo header sockets (six volts), and +9 volts on the carrier board.

Turning It On and Checking Out Operation

I would suggest checking operation in two parts: sonar and IC; then the microcontroller.

(Note: Use proper grounding procedures to eliminate electro-static component destruction.)

Check Sonar and IC Operation

The sonar output and comparator operation should be checked out before the microcontroller is installed. Insert IC U1 (LM339) into the socket. Stand clear from the front of the Max sonar units as they go through a short calibration cycle after turn-on.

Adjust the threshold voltage at the LM339 for about

0.5 volts at pins 5 and 7. With no signal from the sonar units, the outputs on pins 1 and 2 should be low (the output LEDs should be lit). When something is detected by the sonar unit, the output should go high and the LED should go out. You should be able to detect these voltage changes with an oscilloscope or voltmeter. If the LED is turning on too much, you may need to adjust the threshold level to reduce the background noise effects.

Program and Check-Out of the Stamp Operation

If you duplicate this project, you'll need to be somewhat familiar with the Stamp 1 or 2. I learned basic programming with the help of Parallax training manuals and kits. For beginners, I recommend the Parallax basic education books *What's a Microcontroller?* and *Robotics With the BoeBot*. The BoeBot kit is a great starting point for building your first robot (it comes with the book).

Once you are familiar with BASIC Stamp programming, you can try the simple programs in **Figure 7** to see that the microcontroller is working. Carefully plug in the microcontroller board, but don't connect the servos just yet. Download a few short programs to check the Stamp to see if it is working, and then check servo operation.

Program 7a is used to check the operation of the Stamp simply by blinking an LED.

This program tells you that the power supply is working; the microcontroller is working and communicating with the computer editor. If the LED is blinkin,g you can plug in and check the servos.

Program 7b is to confirm that the servos are working. You can check either servo (*PULSOUT* pin 5 or 6). Try changing the *PULSOUT* count (starts at 150) to 100 or 200, or any number in between to see the servo change position.

Eye-Ear Program Code

Figure 7a.

```
'{$STAMP BS1}

' BS1& 2 BOARD WORKING TEST!
' For blinking LED (Program running)
' Connect LED in series with 470 resistor.
' Plus side of LED to Pin zero, resistor side
' to ground.

again:
DEBUG "Hello World", CR
TOGGLE 0
PAUSE 300
GOTO again
```

Figure 7b.

```
'{$STAMP BS1}
' Servo control test
' To test for servo position
' Must re-load program each time pulse number is
' changed to check servo position.


again:

FOR B2 = 1 TO 5          ' repeat five times
  PULSOUT 5, 150         ' pulsout to port 5, PWM
                         ' rate is 150 (change to
                         ' test servo position)
  PAUSE 15               ' wait
  NEXT                   ' goback to 'FOR' 5 times
GOTO again               ' go back to again
```

Figure 7c.

```
'{$STAMP BS1}
'{$PBASIC 1.0}

' Tested 3/11/2013
' Servo Output test
' Test servo range with PWM program
' With BS1 typically PWM of 100 = max.CW,
' 200 max.CCW
' Servo position starts at 100 and counts up
' to 195 then returns to 100.
' You can change the "Start Variable" and/or
' the max count to test the servos range or
' rotation.

'——Declaration———-
SYMBOL counts = W0
SYMBOL Servo = W1

'——-Variables———-
counts = 0
servo = 100
'_______________
Start:
DEBUG "Start"       'Editor prints Start
                    ' Servo going Clock Wise
FOR counts = 1 TO 100     ' Pulse width from
                          ' 1 to 2 ms
  servo = servo +1  ' servo counts + 1 (counts
                    ' up)
  PULSOUT 5, servo  ' output to port 5, servo
                    ' count
  PAUSE 15          ' wait 15ms
  DEBUG W1          ' prints out Servo position
 IF servo => 195 THEN goback
    'if servo position is equal to or greater
    ' than 195
                    ' jump to goback below
 NEXT               ' next pulse width


 goback:            'Servo goes Counter clock
                    ' wise
 DEBUG " goback "   ' Editor print goback
 FOR counts = 1 TO 100    ' for counts 1 to 100
  servo = servo -1  ' servo counts go down (-1)
  PULSOUT 5, servo  'Output port 5, servo count
  PAUSE 15          'wait 15ms
  DEBUG W1          'prints out servo position
  IF servo =<105 THEN start
    ' if servo count is equal to or less than
    ' 105 then Start
  NEXT              'go back to F\OR (until 100)
```

Program 7c will move the servos (step the counts through); first CCW, then CW. You can change the *IF* count value to see the servo position control.

Program 7d is the code for the whole Eye-Ear movement with input detection.

Comments are included and debug statements are used to help you follow the program as it steps through.

Check Eye-Ear operation by holding an object (hand) in front of one of the ears; the eyes should quickly move in that direction. The same is true with the other ear/eye operation.

Going Further

The whole idea of this project was to build something to have a person/robot interaction.

One can use the sonar detectors to perform other types of actions. You could have a hand pointing or a mannekin head that rotates toward a detected object.

Use your imagination and artistic abilities to create an interactive robot that fascinates people and excites children to the field of electronics and robotics. **SV**

Figure 7d.

```
'{$STAMP BS1}
'{$PBASIC 1.0}

'   Eye-Ear 8,  3/5/2013
'   Edit for article
'   Program to detect sonar movement
'   then move and rotate
'   left and right eyes together
'   Added TOGGLE to pin zero, add LED+ R also
'   3/11/2013

  '- - - -Delarations - - - -
  SYMBOL counts = W0
  SYMBOL LeftEye = W2
  SYMBOL RightEye = W3

' - - - Initilazation - - - -
'   variables
  LeftEye = 150
  RightEye = 150
  OUTPUT 0
' - - - - Start Program - - - -
DEBUG CLS
'DEBUG "START PROGRAM   "
PAUSE 3000             ' delay for Sonar self
                       ' calibration
'DEBUG " GO "

Start:        'Start of main program

GOSUB moves
    ' moves subroutine is to output pulses to
    ' right/left eye 10 times

fwd:
DEBUG " fwd  "         ' Forward rotates eyes
                       ' (servos) Clock-Wise
FOR counts = 1 TO 50 ' right & left eyes max
                       ' right CW
righteye = righteye -2
    ' slow step rate counts down by -2
lefteye = lefteye -2
IF righteye =< 100 THEN back
    ' if right eye is equal to or less than
    ' 100 then back
GOSUB moves
' pulse servos 10 times
GOSUB checkin          ' goto checkin subroutine
                       ' to see if input
NEXT                   ' return  to FOR

back:
DEBUG "Reverse "       ' right & left eyes max
                       ' left CCW
FOR counts = 1 TO 50
righteye = righteye +2      ' count + 2
lefteye = lefteye +2
IF righteye => 200 THEN fwd ' if pulsout is
greater than 200 then fwd (CW)
GOSUB moves
GOSUB checkin
NEXT

moves: 'sends pulses to servos x10
FOR counts = 1 TO 10
PULSOUT 6, righteye
PULSOUT 5, lefteye
PAUSE 5
NEXT
RETURN

checkin:
TOGGLE 0         ' blink pin zero (LED), shows
                 ' Stamp operation
' Check for ultrasonic input, Left & Right
INPUT 1
INPUT 2
IF PIN1=0 AND PIN2=0 THEN norm
    ' If normal keep on rotating eyes servo
IF PIN1=1 AND PIN2=0 THEN maxrt
    ' If pin1 high then max right
IF PIN1=0 AND PIN2=1 THEN maxlft
    'If pin 2 high then max left
IF PIN1=1 AND PIN2=1 THEN norm
    'If pin 1 & 2 high then normal rotate

maxrt:           ' sonic sence righteye max right
DEBUG "MAX RIGHT "   ' print out MAX RIGHT
FOR counts = 1 TO 10
righteye = 110       ' right eye is 110
                     ' position
lefteye = 110        ' left eye is 110 position
                     ' also
GOSUB moves
NEXT
PAUSE 100
GOTO back        'eyes servo start to rotate back


maxlft:          'sonic sence lefteye max left
DEBUG "MAX LEFT"
FOR counts = 1 to 10
righteye = 190
lefteye = 190
GOSUB moves
NEXT
Pause 100
GOTO fwd

norm:
RETURN
```

Continued from page 21

ID, position, velocity and Torque control; packet communication over the RS-485 protocol; and a modular approach to build applications.

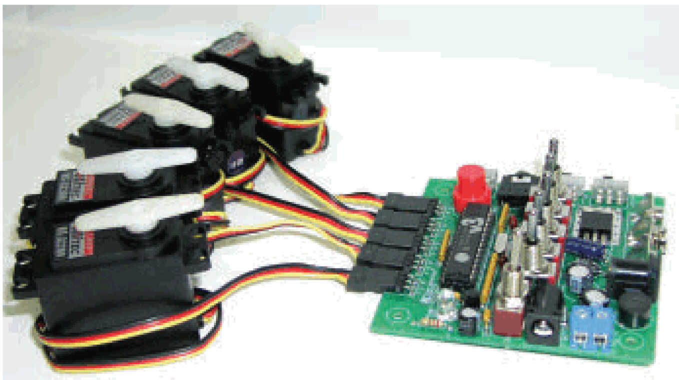
Robot developers spend tremendous amounts of time transforming ideas and designs into reality. As a robot size grows, mechanical complexity increases and demand for an efficient actuation system becomes a top priority. In addition, easy operation and maintenance can be a major bottle-neck to numerous prototype robot designs. ROBOTIS has taken the initiative to solve these problems using its modular design.

For further information, please contact:

Robotis, Inc. Website: **www.robotis.com**

SMC-05 Servomotor Controller

Images Scientific Instrument's updated SMC-05 and SMC-05A servomotor controllers now allow for manual and PC control of five hobby servomotors (Hitec/Futaba). Servomotors are controlled manually via onboard switches, or by a Windows program through a USB port on a PC.



Universal three-position headers make it easy to connect servomotors. Simply plug them into the board.

A PCB allows multiple power supply options for both low and high power servomotor functions.

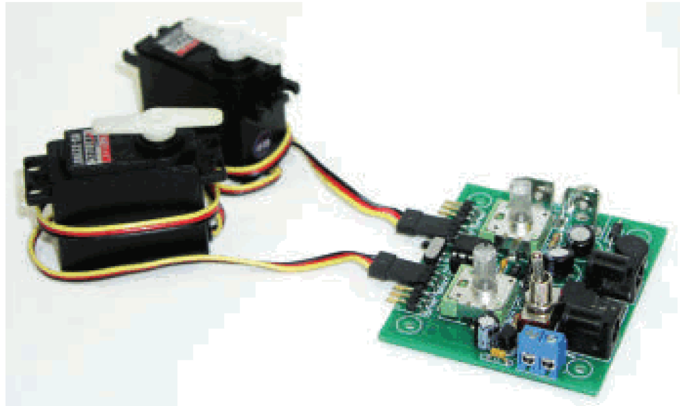
An optional green translucent cover (not shown) is available. Servomotors are not included. Prices are $77.38 for a kit or $92.88 for an assembled unit.

SMC-02 Manual Servomotor Controller

Images Scientific's SMC-02 servomotor controller allows manual control of two hobby servo motors. via onboard potentiometers.

A knob proportionally controls the servomotor shaft. The servo will move as fast and as far as you rotate the potentiometer knob. There are two ports for connecting two external potentiometers to control the servomotors.

Universal three-position headers make it easier to connect servomotors by plugging them into the board.



A PCB allows multiple power supply options for both low and high power servomotor functions.

There is an optional green translucent cover available for this model, as well. Again, the servomotors are not included. Prices are $37.94 for a kit; $52.21 for an assembled unit.

For further information, please contact:

Images, Co. Website: **www.imagesco.com**

Machine-to-Machine Development Platform for Code Division Multiple Access

Microchip Technology, Inc., has announced a new Verizon Wireless Certified Machine-to-Machine

THE OWNERSHIP, MANAGEMENT, AND CIRCULATION STATEMENT OF SERVO MAGAZINE, Publication Number: 1546-0592 is published monthly. Subscription price is $24.95. 7. The complete mailing address of known office of Publication is T&L Publications, Inc., 430 Princeland Ct., Corona, Riverside County, CA 92879-1300.?Contact Person: Larry Lemieux. Telephone: (951) 371-8497. 8. Complete Mailing address of Headquarters or General Business Office of Publisher is T&L Publications, Inc., 430 Princeland Ct, Corona, CA 92879. 9. The names and addresses of the Publisher, and Associate Publisher are:?Publisher, Larry Lemieux, 430 Princeland Ct., Corona, CA. 92879; Associate Publisher, Robin Lemieux, 430 Princeland Ct., Corona, CA?92879. 10. The names and addresses of stockholders holding one percent or more of the total amount of stock are:?John Lemieux, 430 Princeland Ct., Corona, CA?92879; Lawrence Lemieux, 430 Princeland Ct., Corona, CA 92879; Audrey Lemieux, 430 Princeland Ct., Corona, CA 92879. 11. Known Bondholders, Mortgagees, and other security holders: None. 12. Tax Status: Has not changed during preceding 12 months. 13. Publication Title: Nuts and Volts 14. Issue Date for Circulation Data: October 2012-September 2013. 15. The average number of copies of each issue during the proceeding twelve months is:?A) Total number of copies printed (net press run);?9,720 B) Paid/Requested Circulation (1)?Mailed Outside County subscriptions: 4,097 (2) Mailed In-County subscriptions: 0 (3) Paid Distribution Outside the Mail including Sales through dealers and carriers, street vendor, and counter sales and other paid distribution outside USPS:?2,042 (4) Paid Distribution by other classes of mail through the USPS: 0; C) Total Paid Distribution:?6,139; D)?Free or Nominal Rate Distribution by mail and outside the mail (1) Free or Nominal Rate Outside-County Copies: 0 (2) Free or Nominal Rate In-County Copies: 0 (3) Free or Nominal Rate Copies Mailed at other classes through the USPS: 0 (4) Free or Nominal Rate Distribution Outside the mail: 700; E)?Total Free or Nominal Rate Distribution:?700; F)?Total Distribution: 6,839; G)?Copies not distributed:?2,881; H) Total: 9,720; Percent paid circulation: 89.76%. Actual number of copies of the single issue published nearest the filing date is September 2013; A)?Total number of copies printed (net press run) 8,924; B)?Paid/Requested Circulation (1)?Mailed Outside County subscriptions: 4,093 (2) Mailed In-County subscriptions: 0 (3) Paid Distribution Outside the Mail including Sales through dealers and carriers, street vendor, and counter sales and other paid distribution outside USPS:?1,775 (4) Paid Distribution by other classes of mail through the USPS: 0; C) Total Paid Distribution:?5,868; D)?Free or Nominal Rate Distribution by mail and outside the mail (1) Free or Nominal Rate Outside-County Copies: 0 (2) Free or Nominal Rate In-County Copies: 0 (3) Free or Nominal Rate Copies Mailed at other classes through the USPS: 0 (4) Free or Nominal Rate Distribution Outside the mail: 200; E)?Total Free or Nominal Rate Distribution:?200; F)?Total Distribution: 6,068; G)?Copies not distributed:?2,856; H) Total: 8,924; Percent paid circulation: 96.70%. I?certify that these statements are correct and complete. Lawrence Lemieux, Publisher - 10/07/2013.

(M2M) development platform for CDMA. The full-featured platform (developed by Twisthink, LLC — a Microchip Authorized Design Partner) enables custom embedded firmware application development on the PIC32, with local area and remote cellular connectivity. The platform is pre-certified on the Verizon Wireless network, with two-way communication capability. It enables designers to develop on a standard 32-bit platform to collect data, share locally over 10/100 Ethernet and ZigBee®, and communicate to a cloud application.

The M2M development platform includes GPS for location-aware applications; a serial communication interface for simple wired connectivity; microSD card support for code, event, or image storage; and an onboard three-axis accelerometer to monitor motion. Additionally, the platform includes temperature and light sensors, and two expansion ports for custom sensing or connectivity development.

The Verizon Advanced M2M client is included as a binary library in the platform's source code. Verizon's framework allows developers to push the system intelligence out to the edge the network, enabling their designs to work autonomously and only consume data services when needed. The kit also makes use of Verizon's Application Enablement Services (AES), which allows

developers to dynamically change the configuration, monitoring, and thresholds of sensors without re-Flashing the device's application.

Developers can integrate their code with pre-validated communication blocks for remotely monitored and controlled applications. The M2M development platform for CDMA is available for $379.99.

For further information, please contact:

Microchip Technology, Inc.	Website: **www.microchip.com**

by Tom Carroll

Post comments on this article and find any associated files and/or downloads at **www.servomagazine.com/index.php?/magazine/article/november2013_ThenNow**.

Human Augmentation

The media seems to have quieted down a bit about those evil robot drones in the air that are spying on us or killing us at will, and have now shifted to putting robotics in a good light. Exoskeletons as human augmentation have made some recent headlines with the latest Iron Man III film, making over a billion dollars in theaters across the land. Figure 1 shows the film's set with Downey being positioned for a scene. This is human augmentation taken to a whole new level.

Figure 2. The GE HardiMan suit.

The news is filled with heart-warming stories of wounded warriors returning home and made as whole as possible with lightweight exoskeletons that allow them to continue their lives in a more normal fashion.

However, the concept of exoskeletons is not new. In the early 1960s, General Electric developed what was called the HardiMan suit shown in **Figure 2**. Funded by our military, the suit allowed a man to lift 1,500 pounds. The downfall of the technology was that the suit itself weighed 1,500 pounds and required an extensive electronic control system and pressurized hydraulic fluid supply to operate.

GE also built a walking truck-sized robot with power supplied by an onboard engine. These demonstrations were for proof-of-concept only, but ultimately led to today's Boston Dynamics Big Dog and similar robots designed to assist warriors on the battlefield.

The Alien series of films generated a lot of interest in exoskeletons with Sigourney Weaver's character, Ripley and the use of the power loader shown in **Figure 3** to fight the alien queen. In a 2005 article for *SERVO*, I wrote about Monty Reed's pneumatic Lifesuit. A bit later in my November 2009 column, I covered exoskeletons to an extent, but the technology has progressed a lot in the past several years.

People in my robotics group have asked me to dig a bit deeper into what is currently available and on the drawing boards. We think this technology is new, but insects, crabs, and other sea creatures have used external shells driven by internal muscles for millions of years. Control

Figure 1. On the set of *Iron Man 3*.

Figure 3- Sigourney Weaver's power loader in the film, *Alien*.

TWCarroll@aol.com

Figure 4. Robert Downey Jr. and his Iron Man suit.

Figure 5. Scene from Robocop 3.

systems and battery technology have improved to the point that wearable exoskeletons are now possible.

One of the main drawbacks to exoskeleton development has been the dreaded power supply issue — especially for wearable systems. This limitation applies to both disability enabling suits and devices to give able-bodied people extreme capabilities. A human being can carry only so much load on their frame, and a disabled person is even more limited. Exoskeletons of years past have been a bit weighty, and power was derived from NiCad batteries. Reed's Lifesuit uses compressed air from a 3,000 psi scuba tank.

Today's lithium polymer and similar battery chemistries offer high watt-hour capacity (power density) at a lower mass, and more efficient motors are making exoskeletons a reality.

Exoskeletons are not new to experimental and home hobbyist robots, either. Many robots that I have seen that have been built by experimenters use hard external shells for legs and arm sections, and are fastened together by swivel joints. Internal motors or driven cables move these structural pieces much like that of a crab's pincer arms.

Articulated joint fingers for robot hands on experimenter's robots frequently use hollow aluminum or plastic tubing sections with simple swivel joints to hold the pieces together. Internal cables can tug on the individual fingers, allowing them to close in a manner similar to a human hand. I'll discuss that technology in a bit.

Blockbuster Movies Showcase Robotic Exoskeletons

People who understand the applications and theory of robotics have had some trouble with science fiction movies such as the Iron Man series. On one hand, this exposure to the general public of a robot suit that automatically snaps together around the wearer and then rockets into the air for hours at a time certainly gives a very incorrect picture of the state of robotics and rocketry.

For the layman, though, any type of portrayal of robotics that keeps the technology fresh in the minds of the public is good. Most people are certainly aware that a lot of what is shown in these scenarios is not available today, and most likely will not be available in their lifetime.

Real World Ways to Look at Exoskeleton Technology

There are several ways in which to categorize human augmentation with exoskeletons. Quite often, we look at the views presented by films. Though Downey's character — billionaire industrialist Tony Stark — started out with a chest and heart damaged by shrapnel, he was, in all sense of the word, a 'normal' human being with all of his physical capabilities. His artificial heart was nuclear powered. (No comment.)

The suit that he ultimately developed to escape from being captured gave him super human strength, plus a few other capabilities such as the ability to fly. The third movie in the series continues with a similar CGI-produced suit shown in **Figure 4** with all its wondrous capabilities used to save humanity from the evil Mandarin (we're not talking about a small orange here).

As I mentioned previously, the more newsworthy category of augmenting a human is the application of robotics to assist our injured soldiers returning from the battlefield. These individuals once had average or above average physical capabilities, and lost part or most of that due to severe injuries incurred during the Middle East wars.

This same technology is also being applied to persons with disabilities that are not the result of an injury. It is in these applications that our state-of-the-art robotics and mechatronics capabilities are able to develop functional systems to assist people in their daily lives. The 1997 Robo Cop film's main character really falls into this latter category as a severely injured policemen. He was

Figure 6. Evan Ackerman of Dvice.com trying out a HAL suit at CES 2011.

brought back to be a fully-functional cop via an exoskeleton suit with all kinds of bells and whistles. You can see him in **Figure 5**.

The HAL Suit

Stepping aside from fiction and fantasy, I'd like to discuss exoskeleton technology that is actually available today. With a growing elderly population that requires assistance in their homes to live independently and injured service men and women returning from theaters of war, exoskeletons have been a major goal of robotics companies.

As I mentioned, power and motor technology has progressed quite a bit in the past several years, but *control* of an exoskeleton has been difficult until now. In Japan, there is a major thrust in robotics to develop assistive aids for the country's growing elderly population. There are still design issues to solve such as stability for walking, a short time of operation before charging, and user learning curves.

Many people have remarked to me, "What would happen to me if I fell over in that thing? How would I get up?" That issue seems to be one of the main worries about exoskeletons — whether by able-bodied people or those with disabilities.

The Hybrid Assistive Limb or HAL suit has been in the news for several years, and was demonstrated at the 2011 Consumer Electronics Show by Evan Ackerman of **Dvice.com** (refer to **Figure 6**). He only tried the lower portion of the suit and was impressed with the technology.

The HAL suit does not use pressure switches located on the appendages to control the arms and legs. The exoskeleton is controlled by very weak bio-signals that can be detected on the surface of the skin via a series of sensors.

The HAL concept was created by Professor Yoshiyuki Sankai of Tsukuba University in Japan in 1989 after receiving his PhD in robotics. He had long desired to develop a "robotic suit to support humans and expand the physical capabilities of its users, particularly people with physical disabilities."

Development of the HAL Suit

The first part of the development task was to map out the body's neurons that controlled leg movements. By 1993, Sankai and his team had developed a rudimentary working prototype. By the early 2000s, a computer was attached. The complete mechanical system was so bulky that it required a 48 pound battery just to power it. In conjunction with the robotics company Cyberdyne, the suit was improved over the years by losing weight so that it could get by with a much smaller battery.

In 2011, Cyberdyne and Tsukuba University announced that hospital trials of the full HAL suit would begin in 2012, with tests to continue until 2014 or 2015. Two different versions of the system have been developed: the HAL 3 that only provides leg functions; and HAL 5 that is a full-body exoskeleton for the arms, legs, and torso.

The HAL 5 shown in **Figure 7** has the capability of allowing the user to lift and carry about five times as much weight as s/he could lift and carry unaided. **Figure 8** illustrates how signals from a person's brain travel down their legs which, in turn, are picked up by skin sensors that the computer converts to motor driving signals for the exoskeleton's joint motors.

Presently, it is only available for rent in Japan at US$2,000 a month, but should soon become available worldwide. At the end of 2012, over 300 HAL suits were in use by 130 different medical institutions and nursing homes across Japan. In February of this year, the HAL system became the first powered exoskeleton

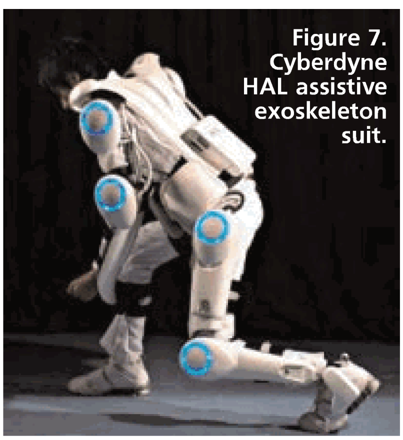


Figure 7. Cyberdyne HAL assistive exoskeleton suit.

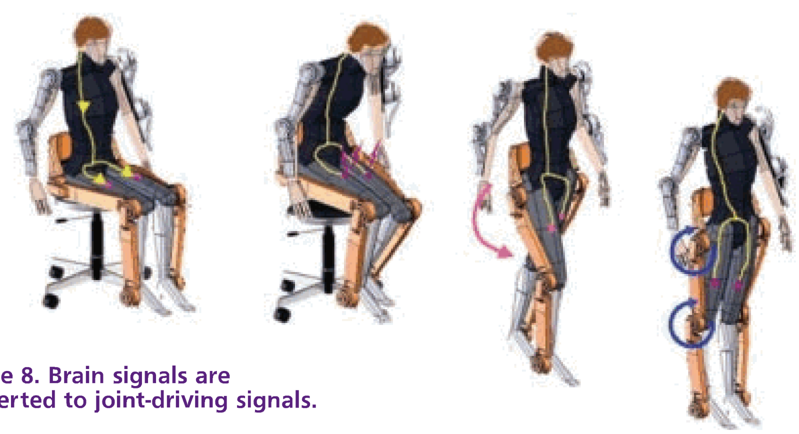


Figure 8. Brain signals are converted to joint-driving signals.

to receive global safety and Underwriters Laboratories certification.

According to Cyberdyne literature:

"HAL senses the weak neuron electrical signals through the wearer's skin that is then fed to an onboard computer. Based on the signals obtained, the power unit is commanded to move the joint in unison with the wearer's muscle movement, enabling HAL to support the wearer's daily activities. This is what we call a 'voluntary control system' that provides movement, interpreting the wearer's intention from the bio-signals in advance of the actual movement. HAL not only has a voluntary control system, but also a 'robotic autonomous control system' that provides human-like movement based on a robotic system which integrally work together with the autonomous control system. HAL is the world's first cyborg-type robot controlled by this unique hybrid system."

The complete robot HAL 5 suit is about 63 inches high — somewhat shorter than the average American — and weighs about 50 pounds. The lower body HAL 3 section is only 33 pounds, though I saw another article that stated 22 pounds for the entire system.

It is designed for daily activities such as standing up from a sitting position in a chair, basic walking, climbing up and down stairs, and holding and lifting heavy objects. It can operate indoors or outdoors for 160 minutes before recharging.

After more clinical trials in Japan, it will be made available in the US and other parts of the world.

Figure 9. Tmsuk rescue dragon robot.

Figure 10. T-52 Enryu robot lifting a car.

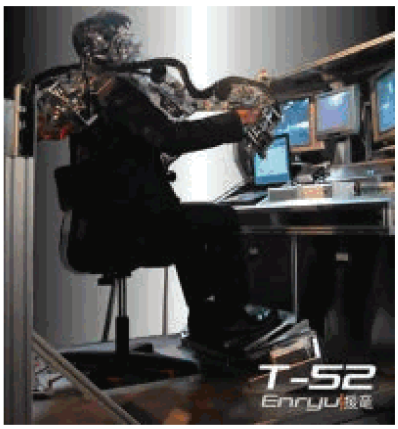


Figure 11. T-52 remote control console.

Figure 12. T-53 Enryu rescue robot.

The Tmsuk T-52 Rescue Dragon

Japan has long been a world leader in specialized robots and the Tmsuk T-52 Enryu "Rescue Dragon" or Hyper Rescue Robot is as amazing as any robot that I've ever read about. For a country plagued by large earthquakes, this massive six-ton monster was developed after the large Kobe earthquake in 1995 for the express purpose of cleaning up large amounts of debris.

This machine is not a true exoskeleton as it is not 'worn,' but it truly augments a human's abilities in delicately handling dangerous electrical lines and hazardous piping.

Standard earth-moving equipment has long been used for this purpose, but the need for large articulated arms to grasp tangled debris, connected timbers, wiring, and plumbing became evident after this disaster. Shown in **Figure 9**, it is easy to see why this massive robot could be called a dragon.

Figure 10 shows the robot removing a car from a snowbank in a 2006 demonstration, though I doubt grabbing it at the edge of the windshield was good for it.

Each of the hydraulically-powered arms can lift 1,100 pounds each. It can be controlled by an onboard pilot or remotely as shown in **Figure 11**.

The Smaller Tmsuk T-53 Enryu

Sometimes a monster like the T-52 is just too big and unwieldy to use in smaller and cluttered debris-filled areas. The more maneuverable T-53 shown in **Figure 12** is still nothing to laugh at as its two articulated arms can handle 220 pounds each. It is much easier to move about the interior of a small building. It was this robot that the disaster relief team at the heavily-damaged Fukushima Daiichi nuclear plant found so useful.

For the longest time, personnel were unsure just what was

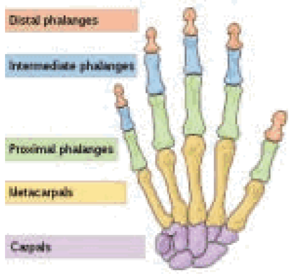


Figure 13. Human hand bone arrangement.

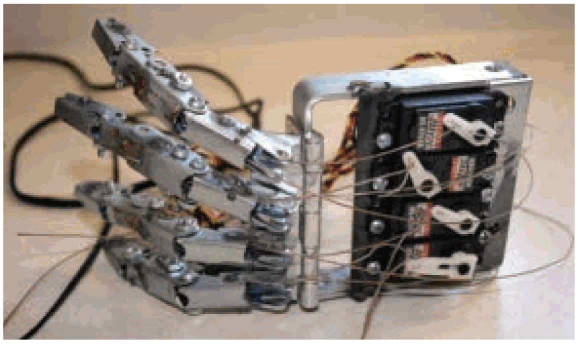


Figure 14. Eric Albers robot hand using RC servos for finger motion.

required to thwart the radioactive side effects from leaking tanks and equipment. The ice wall created by freezing the ground beneath the leaking plant is just a temporary measure; all contaminants must eventually be removed. Many robots from the US (such as PackBots from iRobot) were used in the early cleanup activities.

Authorities then brought in the T-53 that was capable of grasping debris too heavy for the smaller US robots. Operating on tracks like its larger brother, the T-53 could move debris aside to allow personnel in radiation suits to go inside the plant's areas, and assess and clean the areas.

The larger T-52 placed the operator so high that he had to rely on LCD TV panels to see what was happening, whereas the T-53 operator was much closer to the work area.

Radioactive contaminant cleanup as in the Fukushima plant disaster is much more difficult as the operating personnel have to wear protective gear and the robots must be decontaminated after use.

Robotic Appendages for Humans

I'd like to step away from the larger machines and body appliances, and examine what is being developed as *replacements* for lower legs and human hands. Experimenters and amateurs have long been interested in designing and building hands for robots — myself included.

The human hand and its bone structure (refer to **Figure 13)** are fairly complex. The addition of an opposing thumb makes our hand a very useful part of our body, but many robot hand designers choose to not use 'metacarpals' nor an opposing thumb, resulting in a four fingered hand with three segments in each finger. As humans, it is the opposable thumb that allows us to not only pick up objects, but to manipulate them in many ways.

Most animals continually walk on their forelimbs, and their feet are designed for traversing the ground — though claws are sometimes used very effectively to rip open things. Even a simple set of pincers is not very useful for a robot and even less useful for a human, though crabs and lobsters seem to get along okay with them.

Considering the above information, it is natural for robot builders who have designed machines with one or two arms to want to step beyond a simple claw or pincer on the arm's ends. Some of my early hand designs used heavy tygon tubing with slits cut in the sides at the joint areas.

An internal cable pulled from the end of the tubing, causing it to curve inward on the side with the slits. This is a simple method sometimes used by toy robot hands, but continual bending will cause the slits to finally split through.

Much better designs use four

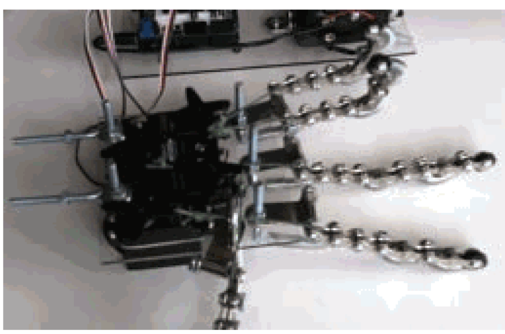


Figure 15. Hackaday.com hardware store robot hand.

Figure 16. iLimb by Touch Bionics.

servos with push-pull rods or cables connected to the four fingers. **Figure 14** is a photo of a robot hand made by Eric Albers. Note the four HiTec servos in the hand's palm (carpal) area.

Another hand from **hackaday.com** is shown in **Figure 15** that is made from easily obtainable hardware store items such as bicycle chain links. These designers and others are not necessarily trying to design a workable hand for a person who has lost their hand(s), but just to advance their robot's capabilities with a very versatile end effector.

iLimb Hand by Touch Bionics

Functional and useful prosthetic hands for use by humans in day-to-day life must pass strict National Institutes of Health and other medical agencies rules and regulations. Ease of attachment to the body is important, as well as comfortable fit for a variety of users and sizes. Ease of use is critical, as well as functionality. These and other goals make human hand augmentation designs a long and drawn out process.

The I-Limb is one of many of these hand prostheses that have been in technology news lately. The i-Limb Ultra prosthetic hand shown in **Figure 16** is designed for those who want more from their prosthesis.

Touch Bionics began in Scotland, and their facilities have spread to the US and the rest of Europe. As with all prosthetic devices, many human clinical trials slowly narrowed down the most effective design. The result is the i-Limb Ultra that provides the ability to gradually increase the strength of its grip on an object.

The assistive device can be controlled via a smart phone as shown in **Figure 17**. This allows instant access to 24 different clasping patterns. These include a wide selection of automated grips and gestures that aid users in completing daily tasks such as using index pointing for typing and a precision pinch mode for grabbing small objects.

The i-Limb hand utilizes pulsing and vari-grip features, allowing the prosthetic hand to implement a firmer hold when conducting tasks such as tying shoelaces or holding a heavy bag securely, or a natural hand position for walking or while at rest.

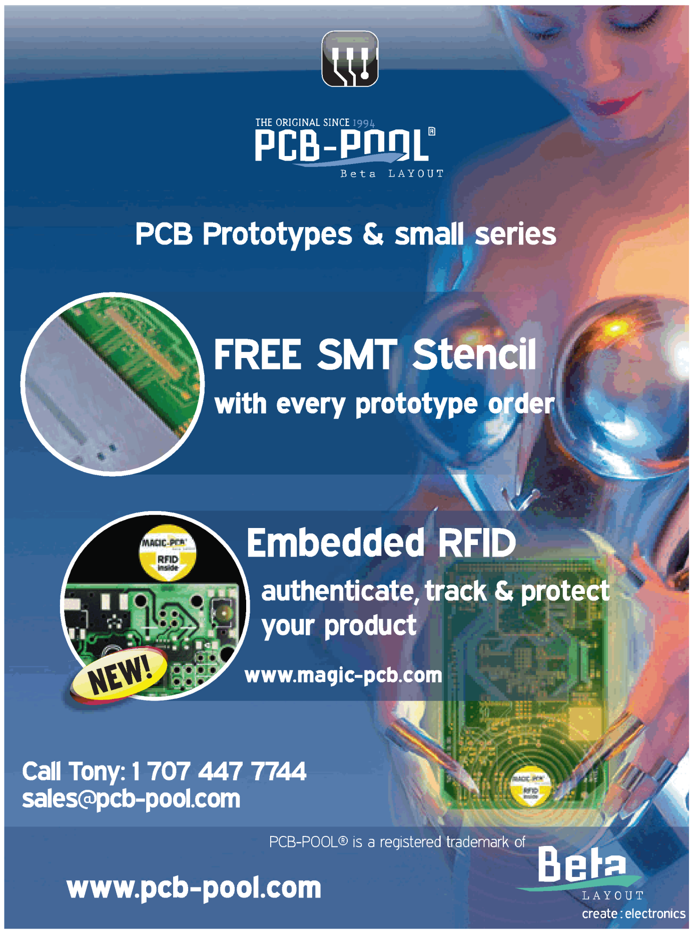
After a period of inactivity, the hand automatically retracts to a natural position. Touch Bionics provides upgraded *biosim-i* and *biosim-pro* control software that allows the hand to be as life-like as possible.

Final Thoughts

This past August, I wrote about robots to serve man. The advances that robotics is giving to medicine with these prosthetic applications is a most vivid example of how the science that we all love is truly serving humankind.

Experimenters in home shops, university labs, and new small businesses will be the catalysts for new and even larger companies that will supply these very personal robotic devices to help change people's lives. Some of you readers will be among these innovators. **SV**

Figure 17. The i-Hand can be configured with a smart phone.



ROBO-LINKS

For the finest in robots, parts, and services, go to www.servomagazine.com and click on Robo-Links.

Advertiser Index

Actobotics™

DREAM. DESIGN. BUILD. REPEAT.

MAJOR PRICE REDUCTION

UNBEATABLE PRICES on SHAFTING & TUBING

1/8" Precision Shafts

Length	Part #	Price
1"	634114	$0.39
2"	634116	$0.49
3"	634118	$0.59
4"	634120	$0.69
5"	634122	$0.79
6"	634124	$0.89
7"	634126	$0.99
8"	634128	$1.09
9"	634130	$1.19
10"	634132	$1.29
12"	634134	$1.39

3/16" Precision Shafts

Length	Part #	Price
1"	634136	$0.49
2"	634138	$0.69
3"	634140	$0.89
4"	634142	$1.09
5"	634144	$1.29
6"	634146	$1.49
7"	634148	$1.69
8"	634150	$1.89
9"	634152	$2.09
10"	634154	$2.29
12"	634156	$2.49

1/4" Precision Shafts

Length	Part #	Price
1"	634158	$0.59
2"	634160	$0.89
3"	634162	$1.19
4"	634164	$1.49
5"	634166	$1.79
6"	634168	$2.09
7"	634170	$2.39
8"	634172	$2.69
9"	634174	$2.99
10"	634176	$3.29
12"	634178	$3.59

1/4" Precision D-Shafts

Length	Part #	Price
1"	634058	$0.99
1.25"	634060	$1.09
1.375"	634062	$1.19
1.50"	634064	$1.29
1.75"	634066	$1.39
2"	634068	$1.49
2.25"	634070	$1.59
2.375"	634072	$1.69
2.5"	634074	$1.79
2.75"	634076	$1.89
3"	634078	$1.99
4"	634080	$2.19
5"	634082	$2.49
6"	634084	$2.89
7"	634086	$3.19
8"	634088	$3.49
9"	634090	$3.79
10"	634092	$4.19
12"	634094	$4.69

3/8" Aluminum Tubing

Length	Part #	Price
2"	635240	$0.99
4"	635242	$1.59
6"	635244	$2.19
8"	635246	$2.79
10"	635248	$3.39
12"	635250	$3.99

1/2" Aluminum Tubing

Length	Part #	Price
2"	635252	$1.49
4"	635254	$2.09
6"	635256	$2.69
8"	635258	$3.29
10"	635260	$3.89
12"	635260	$4.49

5/8" Aluminum Tubing

Length	Part #	Price
2"	635264	$1.99
4"	635266	$2.59
6"	635268	$3.19
8"	635270	$3.79
10"	635272	$4.39
12"	635274	$4.99

5/16" Precision Shafts

Length	Part #	Price
1"	634180	$0.69
2"	634182	$1.09
3"	634184	$1.49
4"	634186	$1.89
5"	634188	$2.29
6"	634190	$2.69
7"	634192	$3.09
8"	634194	$3.49
9"	634196	$3.89
10"	634198	$4.29
12"	634200	$4.69

3/8" Precision Shafts

Length	Part #	Price
1"	634202	$0.89
2"	634204	$1.39
3"	634206	$1.89
4"	634208	$2.39
5"	634210	$2.89
6"	634212	$3.39
7"	634214	$3.89
8"	634216	$4.39
9"	634218	$4.89
10"	634220	$5.39
12"	634222	$5.89

1/2" Precision Shafts

Length	Part #	Price
1"	634224	$1.29
2"	634226	$1.99
3"	634228	$2.69
4"	634230	$3.39
5"	634232	$4.09
6"	634234	$4.79
7"	634236	$5.49
8"	634238	$6.19
9"	634240	$6.89
10"	634242	$7.59
12"	634244	$8.29

1.0" Aluminum Tubing (Flanged)

Length	Part #	Price
2.125"	635176	$2.99
2.250"	635166	$3.49
3.375"	635178	$4.49
4.725"	635180	$4.99

1.0" Stainless Steel Tubing

Length	Part #	Price
2"	635134	$2.59
3"	635136	$3.39
4"	635138	$4.29
5"	635140	$5.19
6"	635142	$6.19
8"	635146	$8.19
10"	635150	$10.19
12"	635152	$11.99

1.0" Aluminum Tubing

Length	Part #	Price
2"	635102	$1.29
3"	635104	$1.39
4"	635106	$1.89
5"	635108	$2.29
6"	635110	$2.59
8"	635114	$2.99
10"	635118	$3.49
12"	635120	$4.49
14"	635122	$5.39
16"	635124	$6.69
18"	635126	$8.19
24"	635132	$11.49

HUNDREDS OF ACCESSORIES

WE HAVE THE PARTS FOR YOUR IDEAS!

- $2.49 ea. 1/4" Shaft Clamps
- $5.99 ea. Tube Clamp Hubs
- $5.99 ea. 1" Bore Tube Clamp
- $8.99 ea. 90° Tube Clamp Hub
- $14.99 ea. Clamping Hub C
- $5.99 ea. Clamping Hub A & B
- $7.99 ea. 5/8" Bore Tube Clamp
- $8.99 ea. 1" Bore Tube Clamp
- Spacer $1.69 / 12 Plastic Shaft Spacers
- $4.99 ea. Set Screw Couplers
- $5.99- $7.99 ea. 90° PillowBlocks
- $0.46- $1.74 ea. Set Screw Collars
- $5.99- $7.49 ea. Bearing Mounts
- $8.49- $9.99 ea. 1" Bore PillowBlocks
- $1.99 ea. Steel Shim
- $6.49 ea. Hub PillowBlocks
- $1.69 - $7.99 ea. Precision Ball-Bearings
- $4.99 - $7.99 ea. Clamping Collars
- $4.99 ea. 1/4" Shaft Coupler
- $6.99 - $7.99 ea. Quad PillowBlocks
- $12.99 ea. Servo Shaft Couplers
- $14.99 - $19.99 ea. Universal Joints
- $12.99 ea. Shaft Couplers
- $6.99 ea. Dual Ball Bearing Hub
- $7.99 ea. Clamping Hubs
- $4.99 ea. Set Screw Hubs
- $2.99 - $15.99 ea. Aluminum Channel
- $12.99 ea. Pinion Gears

Browse our entire selection online!

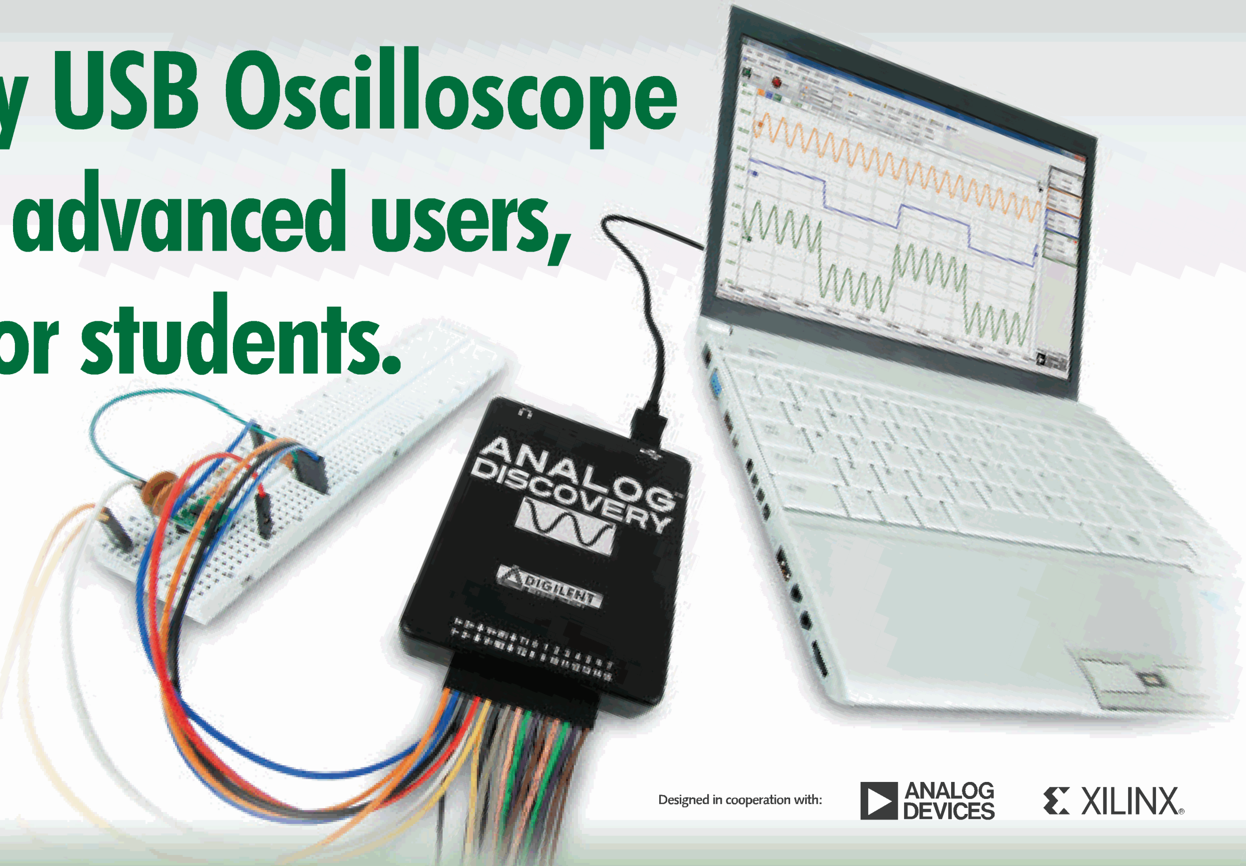
JOIN OUR ONLINE COMMUNITY!

Shop Online | Call 620.221.0123 | sales@servocity.com

www.SERVOCITY.com

Copyright 1999 - 2013 Robotzone, LLC - All rights reserved. ServoCity™ and Actobotics™ are registered trademarks of Robotzone, LLC. Pricing & availability subject to change.

The only USB Oscilloscope built for advanced users, priced for students.



$99.00
(U.S. student price)

Designed in cooperation with: ANALOG DEVICES XILINX

Turn any PC into a powerful electrical engineering workstation! The USB-powered Analog Discovery lets you measure, visualize, analyze, record and control mixed signal circuits of all kinds. It's small enough to fit in your pocket, but powerful enough to replace a stack of lab equipment. Driven by the free WaveForms™ software, the Analog Discovery lets you build and test analog and digital circuits in virtually any environment, in or out of the lab.

- 2-Channel Oscilloscope
- 2-Channel Waveform Generator
- 16-Channel Logic Analyzer
- 16-Channel Digital Pattern Generator
- ±5VDC Power Supplies
- Spectrum Analyzer
- Network Analyzer
- Voltmeter
- Digital I/O

Now supported by MATLAB / MATLAB student edition

Also Available:

Analog Parts Kit (APK)

- 20+ ICs from Analog Devices
- 200+ discrete components
- Solderless breadboard
- Perfect for Circuits 1&2 classes

$69.99

Discovery BNC Adapter

- Allows the use of standard BNC-terminated probes
- Selectable AC & DC coupling

$19.99

DIGILENT® BEYOND THEORY

www.digilentinc.com/AnalogDiscovery

ANALOG DISCOVERY™ USB OSCILLOSCOPE+